MASTER IN COMPUTER SCIENCE

Network management in an interconnected networking environment : The current state of art from the OSI point of view and a practical implementation

Detrembleur, Bernard

Award date:
1987

Awarding institution:
University of Namur

Année Académique 1986 - 1987

NETWORK MANAGEMENT IN AN INTERCONNECTED NETWORKING ENVIRONMENT:

The current state of art from the OSI point of view and a description of a practical implementation.

Bernard Detrembleur

Directeur : Philippe van Bastelaer

Mémoire présenté
en vue de l'obtention
du titre de
licencié et maître
en Informatique

RUE GRANDGAGNAGE,21,B - 5000 NAMUR (BELGIUM)

Acknoledgments

I would like to thank all the people who have contributed to the elaboration of this work.

First, Mr. Philippe van Bastelaer, my work director at the Facultés Universitaires Notre Dame de la Paix, Namur, who offered me the opportunity of a training at the European Center for Nuclear Research (CERN), Geneva, during the first semester of the academic year 1986 - 1987.

Then, Mr. Bob O'Brien, my training director at the European Center for Nuclear Research, Geneva, who introduced me to the problem of network management in an heterogeneous environment.

And finally, all the members of the Data Division / Communication Section (DD/CS) of the CERN, with whom I spent six unforgettable months. They are:

- Brian Carpenter
- François Fluckiger
- Bob O'Brien
- Denise Heagerty
- Giorgio Heiman
- Olivier Martin
- Kik Piney
- Ben Segal
- Achille Petrilli
- Mike Gerard
- Marie-Thérèse Monnet
- Yves Grandjean
- Christian Isnard
- Danny Davids
- Eric Julien

The practical experience gained with them was an inevaluable source of ideas for this work.

I also thank all the people I have forgotten and who have helped me in this work.

Contents

FIGURES:

1. Introduction

The aim of this work is to present the current state of the art in the network management. We will take two different points of view: the first is an academical point of view through the present level of standardization on this subject. Therefore, we will present the seven layer model of the International Standard Organisation (ISO) and then the management framework presented by ISO. The second is a practical point of view through a multi-network management system developed by the European Center for Nuclear Research (CERN) in Geneva: the Common Operational Management System (COMS). This will form the skeleton of the work: chapter 1 is the introduction, chapter 2 is a brief overview of the seven layer model, chapter 3 is a presentation of the interconnection of open systems, chapter 4 is the ISO point of view about network management, chapter 5 is the presentation of the COMS project and chapter 6 is the conclusions of the work.

The starting point of our work on the subject is a training made in CERN during the second semester of 1986. During this training, we have collaborated with the COMS team.

This introduction contains some remarks on the relations between ISO and OSI, their definitions, a presentation of the CERN and the aim of the COMS project.

1.1 Definitions of ISO and OSI

The International Organization for Standardization is an international institution based in Geneva grouping all the national committees for standardization. The application field of ISO is very wide (e.g. from the heigth of the bumpers to the architecture of computer networks.) ISO groups institutions like AFNOR (Association Française de Normalisation) or ANSI (American National Standards Institute). In 1961, ISO created a technical committee (TC97) on "Computers and Data processing". Then in 1977, ISO recognized the urgent need for standards for heterogeneous computer networks. They decided to create a new subcommittee (SC 21) on Open Systems Interconnection (OSI).The scope of TC 97 / SC 21 is "the development of a schematic model for system architecture for open systems working, identifying and specifying interfaces which should be considered for standardization. [PIA80]" A document proposed by a subcommittee is a Draft Proposal (DP); after the vote of the different countries involved in the subcommittee, it becomes a Draft International Standard (DIS) and it is presented to the Technical Committee to become an International Standard (IS). It is evident that during this evolution the documents are subject to changes, the reader will refer to the status of the references (DP,DIS or IS) to see the level of finition.

Figure 1.1 illustrates the links between the different parts of ISO and the document they produce.

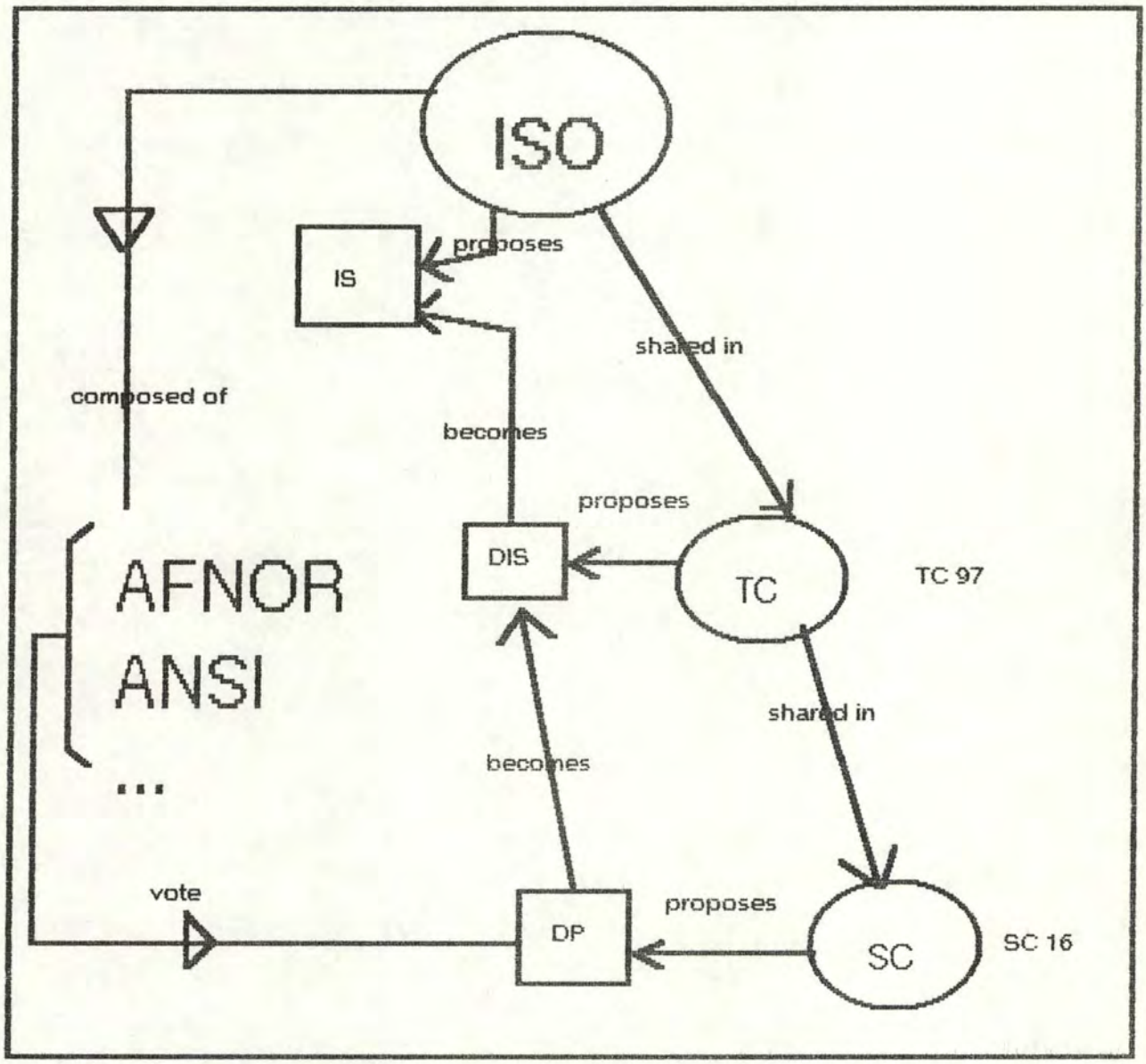


Figure 1.1: Organization of ISO.

1.2 Warning about ISO

ISO is an international institution. For this reason, it must be very wide in its recommendations and is under the domination of large countries with divergent opinions depending on their national interest.

Furthermore, all the most important network manufacturers had their own architecture before the edition of the seven layer model of ISO.

This will reduce the practical influence of ISO recommendations.

1.3 The CERN and the COMS project

The CERN is the European Council for Nuclear Research. The aim of the different laboratories is to provide particle accelerators to the European searchers.

This explains the heterogeneity of the computer systems in CERN. All the teams coming in CERN for experiences, have to take the results with them for further analysis. But they all have different computers or different ways of communication. This involves that the CERN needs to have a very wide range of computers and networks.

All this infrastructure must remain operational as long as possible. For this reason, CERN has decided to develop a management project to monitor and manage all the CERN networks. This is the aim of the COMS project.

2. The OSI basic reference model

2.1 Historic of OSI

In 1977, ISO recognized the urgent need of standards for heterogeneous informatic networks and decided to create a new subcommittee (SC21) on "Open Systems Interconnection (OSI)". Before 1977, each manufacturer developed his own architecture to interconnect his own equipment. But the need of interconnecting equipments from different manufacturers became evident, this need was starting event of OSI. From the OSI point of view, "open" means that conformable to the international standards, a system will be capable of interacting with all other systems obeying to the same standards. So the fact that a system is open, refers to the mutual recognition of two open systems and does not imply any particular implementation or technology. In 1979, the results of the work of SC21 became a draft international standard (DIS), known as DIS 7498: Basic reference model. In 1984, DIS 7498 became an international standard (IS), known as IS 7498: Basic reference model.

2.2 Definitions, abbreviations and notations

The basic reference model introduces new concepts. These concepts will be defined and we will give the usual abbreviations for OSI and finally some notations.

2.2.1 Definitions

All the definitions given now, refer to the OSI basic reference model [I7498].

Real system: A set of one or more computers, the associated software, peripherals, terminals, human operators, physical processes, information transfer means, etc..., that forms an autonomous whole capable of performing information processing and/or information transfer.

Real open system: A real system which complies with the requirements of OSI standards in its communication with other real systems.

Open system: The representation within the OSI Reference Model of those aspects of a real open system that are pertinent to OSI.

(N)-subsystem: An element in a hierarchical division of an open system which interacts directly only with elements in the next higher division or the next lower division of that open system.

(N)-layer: A subdivision of the OSI architecture constituted by subsystems of the same rank (N).

(N)-entity: An active element within an (N)-subsystem.

Peer entities: Entities within the same layer.

(N)-services: A capability of the (N)-layer and the layers beneath it, which is provided to (N+1)-entities at the boundary between the (N)-layer and the (N+1)-layer.

(N)-protocol: A set of rules and formats (semantic and syntactic) which determines the communication behaviour of (N)-entities.

(N)-service-access-point: The point at which (N)-services are provided by an (N)-entity to an (N+1)-entity.

2.2.2 Abbreviations

(N)-SAP: is (N)-service-access-point.

2.2.3 Notations

In the next sections, we will analyze the layers. The following notation is used to identify the different layers:

* (N)-layer means any specific layer.
* (N - 1)-layer means the next lower layer.
* (N+1)-layer means the next higher layer.

2.3 Principles of layering

"Layering is a structuring technique which permits the networks of Open Systems to be viewed as logically composed of a succession of layers, each wrapping the lower layers and isolating them from the higher layers [GRE83]." Each layer will be defined by the specific set of functions it performs. The independence of the layers is ensured by defining the services provided by (N)-layer to the (N+1)-layer, independently of how these services are performed. This approach allows to change the way a layer or a set of layers operate, provided they still offer the same service to the (N+1)-layer. Each layer can be seen as a black box where only the interfaces (the services provided) are known. This technique is similar to the one used in structuring programming where only the functions performed by a module are known by the user (human or another module).

2.4 The seven OSI layers

2.4.1 Overview

OSI has applied the principles of layering for the network architecture. A system is composed of subsystems corresponding to the intersection of the system with a layer. A layer consists of subsystems of the same rank and a subsystem is made of entities. Figure 2.1 gives a graphical representation of the notions of system, layer, subsystem and entity.

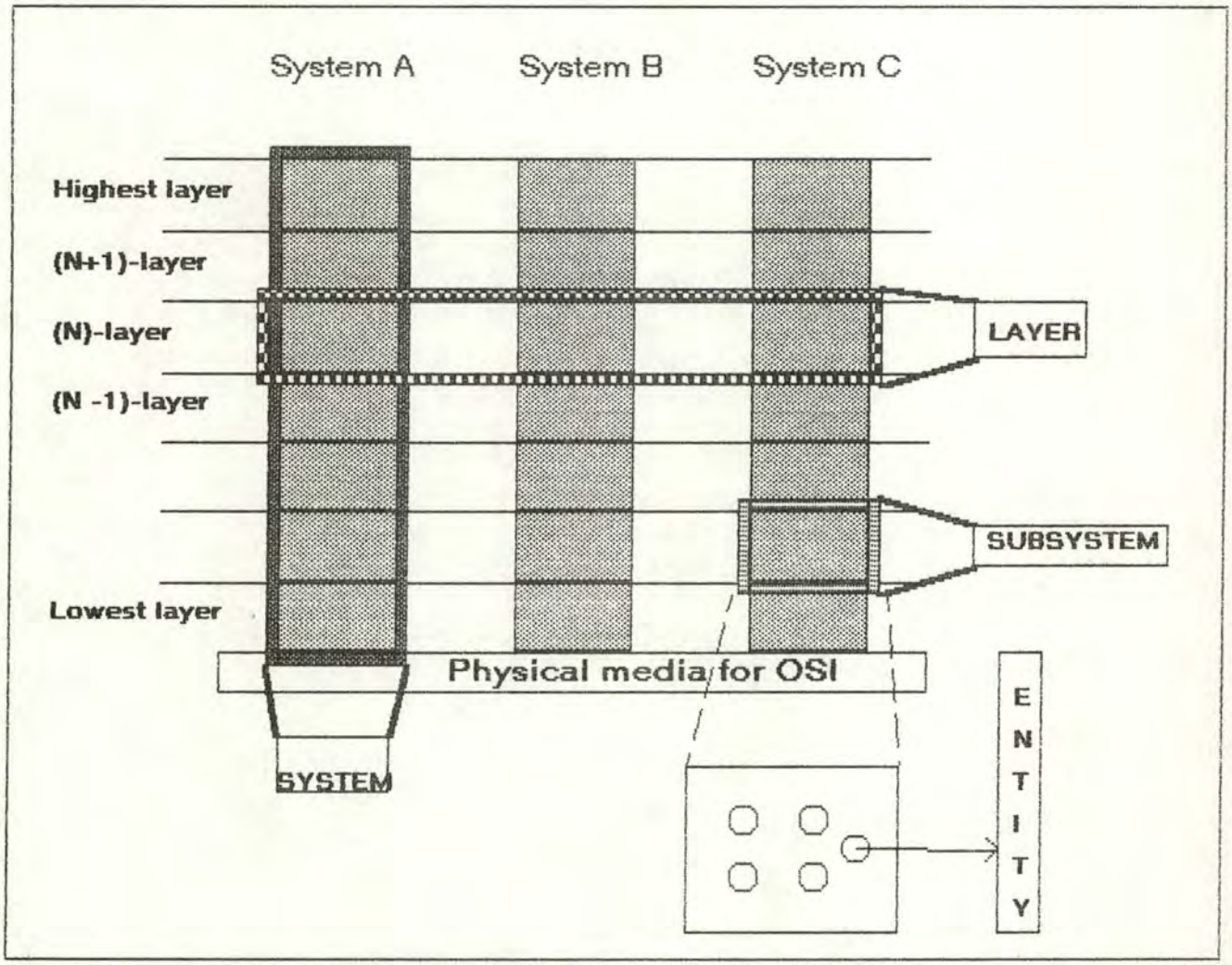


Figure 2.1: System, layer, subsystem and entity.

Entities of the same layer are called peer entities.Each layer (except for the highest layer) works to provide the (N)-service to the (N+1)-layer. It means the (N)-entities add value to the (N-1)-service they get from the (N-1)-layer and they offer this added-value (the (N)-service) to the (N+1)-layer. Communications between the (N)-entities is made exclusively by the use of (N-1)-services. The communication between the (N)-entities is ruled by the **(N)-protocol** which defines precisely how the (N)-entities work together using the (N-1)-service to perform the (N)-service provided to the (N+1)-layer. The (N)-service is provided at (N)-service-access-point. Figure 2.2 gives a graphical representation of the concepts of service, protocol and SAP.

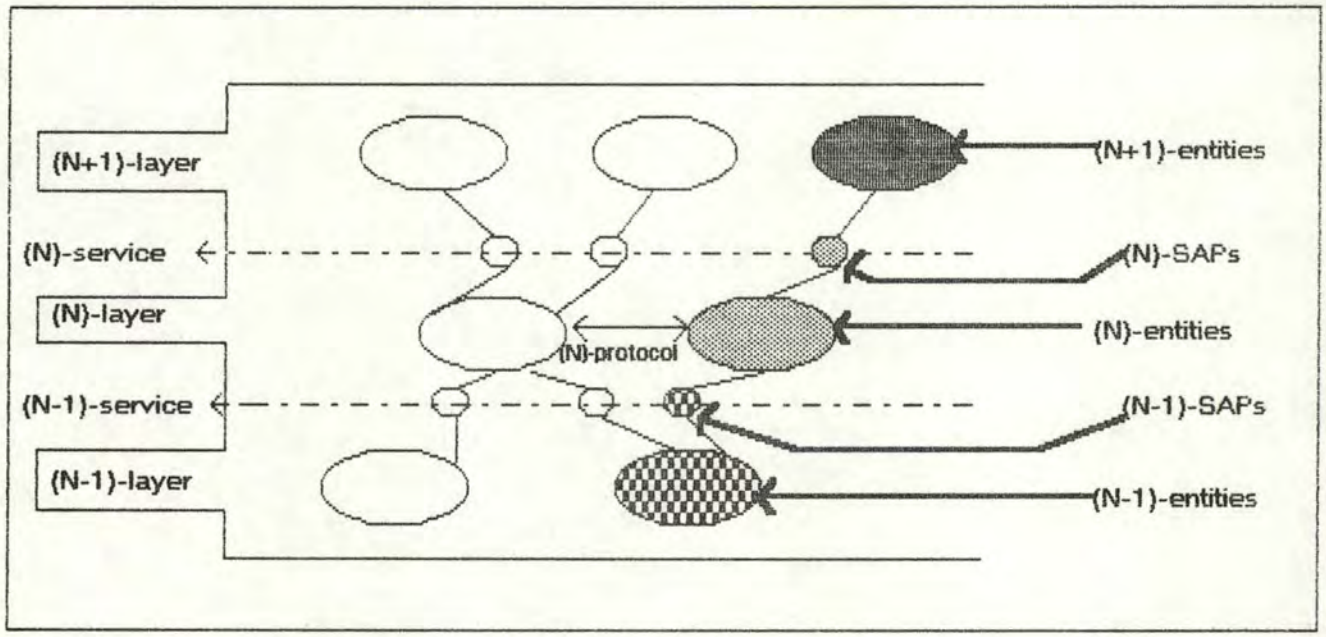


Figure 2.2: Service, protocol and SAP.

2.4.2 Principles used to determine the seven layer

To create the different layers of its model, OSI has respected some general principles.

-The number of layers must be limited to prevent difficulties when describing or integrating the layers.

-A new boundary can be created when the description of the service is small and the number of interactions across the boundary is minimized.

-A layer contains similar functions.

-A new layer is created when a different level of abstraction is needed in the handling of data.

2.4.3 The physical layer

"The purpose of the physical layer is to provide mechanical, electrical, functional and procedural means to activate, maintain and de-activate physical connections for bit transmission between data-links entities [I7498]". In practice, it means that the aim of the physical layer is to transmit bits at the right speed with a reasonably low rate of error from a A site to a B site. The physical layer will form a data circuit between the two sites. Figure 2.3 shows the data circuit between the two sites, DCE (data circuit terminating equipment) is usually a modem.

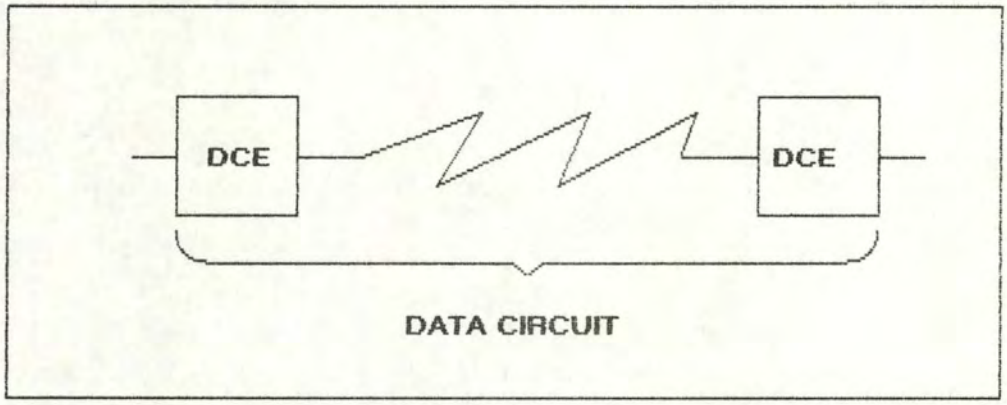


Figure 2.3: Data circuit.

The services provided are the bit transmission, the initialization of the circuit by the sending site, the wake-up of the receiving site and the closing of the circuit.

2.4.4 The data link layer

"The purpose of the data link layer is to provide functional and procedural means to establish, maintain and release data link connections among network entities. It detects and possibly corrects errors which may occur at the physical layer [I7498]." So the aim of the data-link layer is the error handling and the flow control. The errors can occur for two main reasons: the imperfection of the modems (DCE) and the parasitic elements. Figure 2.4 shows the data link between two physical layers connected by a data circuit.

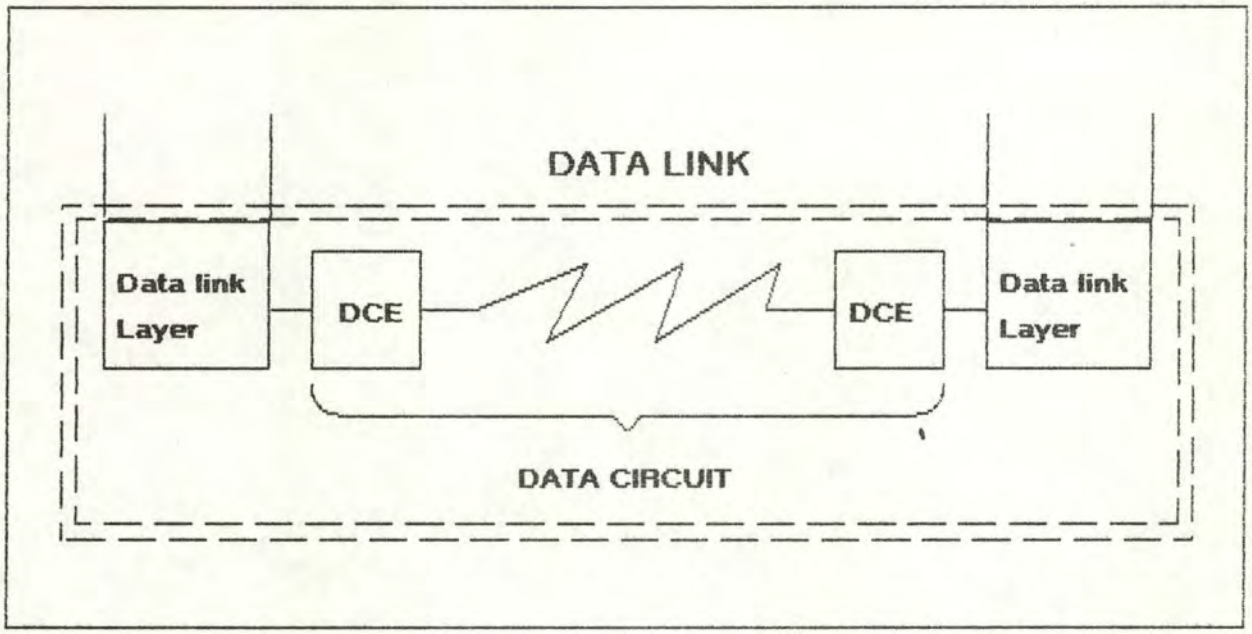


Figure 2.4: Data link.

For the Local Area Networks, the layer 2 has been divided into two sublayers: the Logical Link Control (LLC) and the Medium Access Control (MAC).The LLC will manage the flow control and the error handling; it is independent of the sharing algorithm which is managed by the MAC. For LANs, the LLC layer is usually a procedure nearly similar to HDLC and the MAC layer is CSMA/CD for Ethernet, token ring for the token ring and token bus for the token bus.

2.4.5 The network layer

"The network layer provides to the transport layer independence from routing and relay considerations associated with the establishment and operation of a given network connection.[I7498]" So the basic service of the network layer is the transparent transfer of data between transport entities by ensuring the routing and the relaying through different interconnected networks.
The network layer is separated into three sublayers: the subnetwork independent convergence role, the subnetwork dependent convergence role and the subnetwork access role with the different protocols associated for each of the sublayers.

The subnetwork independent convergence protocol (SNICP) operates to construct the OSI Network Service in concordance with a set of assumptions about the underlying service available. These assumptions reflect choices made about the network topology, the technical and economic advantages and disadvantages associated with assuming one type of underlying service rather than another and the degree of complexity of protocols mechanisms that can justifiably be sustained by the protocol.

The subnetwork access protocol (SNACP) operates under constraints explicitly stated as characteristics of a specific subnetwork. The sublayer contributes to the provision of a subnetwork service which is specific to the subnetwork concerned.

The subnetwork dependent convergence protocol (SNDCP) operates over a protocol providing the SNACP role. It is used to provide the service assumed by a protocol fulfilling the SNICP role or to provide the Network Service directly. So a SNDCP is constrained by the specific characteristics of a particular subnetwork service (SNACP) and by the assumptions made by an SNICP. This layer allows to increase the generality and the simplicity of SNICP by enabling the reduction of the constraints induced by many SNACP on an SNICP.

Figure 2.5 shows the internal subdivision in three sublayers of the network layer.

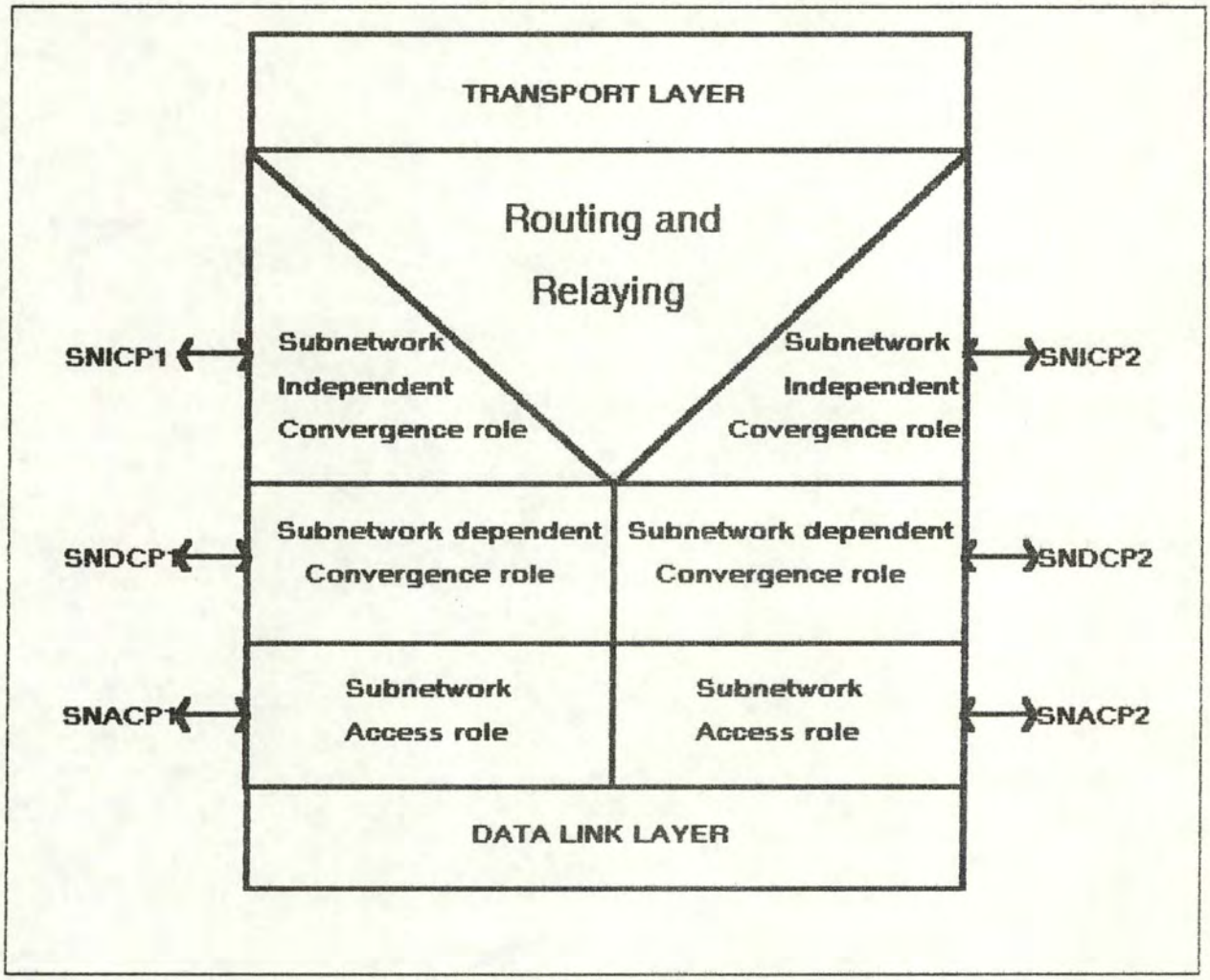


Figure 2.5: The network layer.

These different sublayers will be used in chapter 3 when treating about the interconnection of networks.

2.4.6 The transport layer

"The aim of the transport layer is the transfer of data between session entities and relieve them from any concern with the detailed way in which the reliable and cost effective transfer of data is performed [I7498]." In fact over this layer, there will be no more problems of error handling. But the kind of services required can be different according to the use. For example, a user may want to have no error rate smaller than 10^{-9}, therefore, OSI has defined different classes of services. Figure 2.6 shows the four different classes of services available. For example, TP4 means Transport Protocol with recovery on outstanding errors, multiplexing and flow control, and recovery on not-outstanding errors.

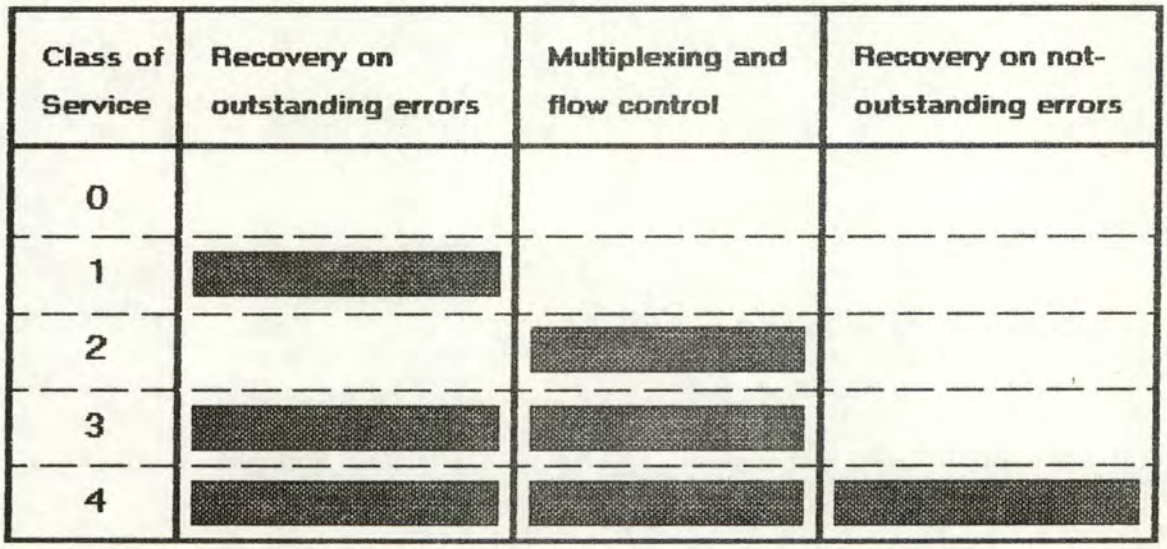


Figure 2.6: Transport classes

2.4.7 The session layer

The session layer is responsible for all the chronological aspects. The session layer provides to the upper layers tools to organize and to synchronize the dialogue of two presentation entities and to manage their data transfer. To implement the data transfer between the presentation entities, the session connection is mapped onto and uses a transport connection.

2.4.8 The presentation layer

The purpose of the presentation layer is to provide to the application layer a representation of information that application entities communicate or refer to in their communication. There are two aspects of the representation of information:

-the representation of data to be transferred between application entities.

-the representation of the data structure which application entities refer to in their communication, along with the representation of the set of actions performed on this data structure.

This means that the presentation level is only concerned with syntax : the representation of the data, and not with semantics: the meaning of the data.

2.4.9 The application layer

The application layer allows application entities to access the OSI environment. So this layer is a window between correspondent application processes which are using OSI for the exchange of meaningful information. The services provided by the application must be directly usable by the application processes or by the user.

2.5 Reasons for the choice of the layers

The physical layer was identified to allow the use usage of a realistic variety of physical media for interconnection with different control procedures (V24,V25,...)

The data link layer was identified for the following reasons: Some physical communication media require special techniques to transfer bit despite a high error rate (e.g. the telephone line).These techniques are known for a long time as data link control procedures. A new physical media (e.g. optical fiber) will require different data link control procedures.

Some open systems will act as a final destination and others will act only as an intermediate node. This leads to the creation of a network layer to provide a connection path to higher level.

The control of data from source end open system to destination open system is the last function to perform in order to provide a complete transport service. This function is the work of the transport layer.
There is a need to organize and to synchronize dialogue, and to manage the exchange of data. That leads to the identification of the session layer.
There is need for general functions related to the representation and the manipulation of data for the benefit of the application programs. This is done by the presentation layer.
There are applications consisting of application processes which perform information processing. This is the job of the application layer.

2.6 Evaluation of the model

This model is a very good abstract model, it uses all the power of the layering and the abstraction induced by the "utilize" relation between the different layers.
Until recently, it was only a reference model, but it seems now that there is a will to develop implementations of full-OSI networks (e.g. Digital, Bull, IBM, TOP-MAP, OSITOP,...). But a few problems have appeared.

- Nothing is defined on the way the communication between layers is established, this induces that it is nearly impossible to have layers from different producers.

- The layers one to four are fully defined. But the layers five to seven are still in a draft form.

- The layered architecture seems to induce some problems of redundancy. Each layer adds a header and a trailer to the message. This gives for a full-seven-layers architecture a very bad rate of effective information in the message.

But, all these problems will certainly be solved in the next years.

3. The interconnection of networks for OSI

3.1 Introduction

We will show in this chapter the complexity of the network interconnection in a theorical point of view. The high level of complexity will create the need of a system to manage all the open systems interconnected or a part of them.
The seven layer model is designed to facilitate the interconnection of networks and it helps, but the solutions are not trivial due to necessary wideness of the model to permit a large heterogeneity of the networks.
We will show different scenarios of interconnection for layer three (network layer). The layers two, four and seven are also layers used for the interconnection. But it is not the aim of this chapter to be complete on the problem of network interconnection, we just want to give the reader an idea of its complexity.
Gateways at level three will be called in this chapter network relays.
We will present some definitions, some concepts and terminology about the network layer, we will again explain briefly the internal organization of the network layer and then we will present the interconnection and some scenarios.
This chapter is based on the ISO norm ISO/DIS 8648 [I8648].

3.2 Definitions

Real subnetwork: a collection of equipment and physical media which forms an autonomous whole and which can be used to interconnect real systems for purposes of communication.

Subnetwork: an abstraction of a real subnetwork.

Interworking unit: one or more items of equipment, or a part of an item of equipment, whose operation allows interconnection of a real subnetwork and another real world component.

Relay system: an abstraction of the equipment forming an interworking unit.

Intermediate system: an abstraction of a real system providing a network relay function, that is a real system which receives data from one correspondent network-entity and forwards it to another correspondent network-entity.

3.3 Concepts and terminology about the network layer

OSI works in an abstraction world. For this reason, we will give here a correspondance between the objects of the real world and the objects in the OSI world. The whole picture will be done with the representation of the system for the real world in the upper part of the picture and the OSI representation in the lower part of the picture. The links between the objects of the real world and the objects in the abstract world will be done with dotted lines.
We will present the concept of end systems and intermediate systems, the concept of real subnetworks and subnetworks and the concept of relay systems and interworking units.

3.3.1 End system

An end system can communicate with another end system either:

a- directly, without an intervening intermediate system

b- through one or more intervening intermediate system.

From this point of view, "directly" refers to that communication which involves only the network entities in the two communicating end systems. No constraints are placed on the way in which relays at lower layers might be used to provide communication between network entities. Figure 3.1 shows the communication between two end systems. The picture contains an example of direct communication and an example of communication through an intermediate. In each of both parts of the picture the real world object are represented upper and the abstract representation of the object are shown lower.

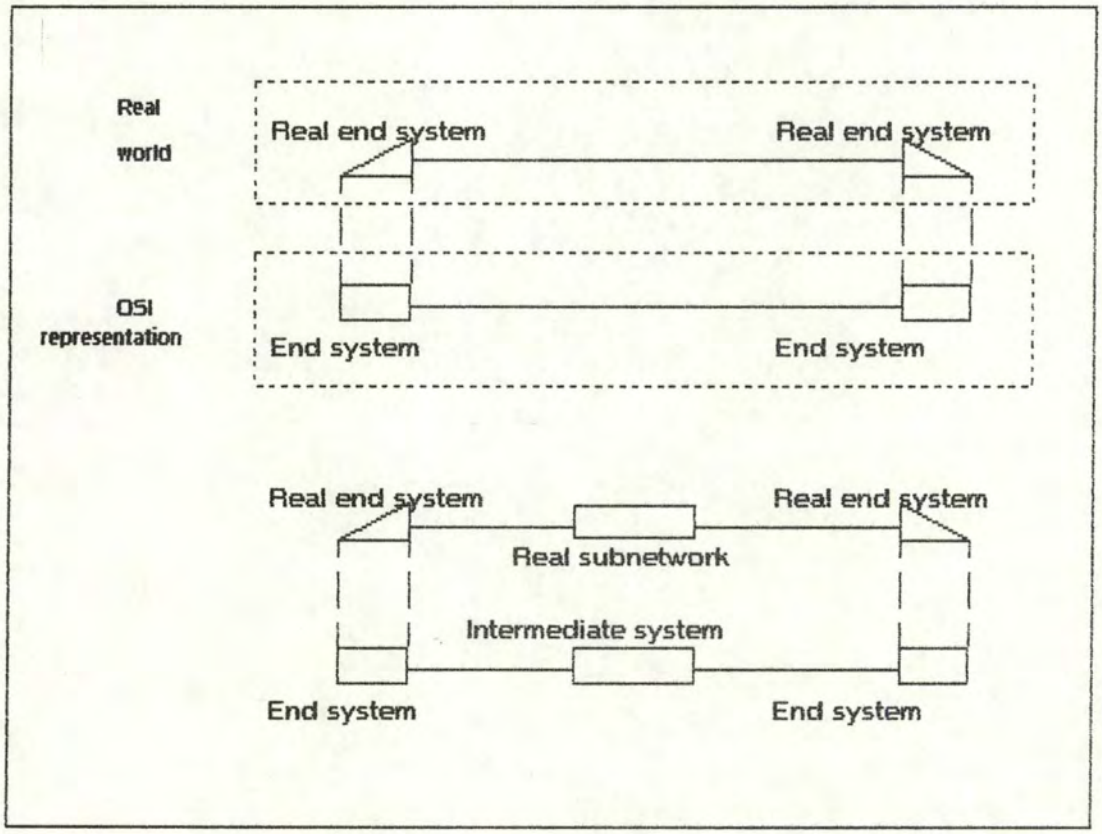


Figure 3.1: End system communication.

3.3.2 Intermediate system

An intermediate system is an abstraction of :

1- A real subnetwork.
2- An interworking unit, connecting two or more real subnetworks.
3- A real subnetwork with an associated interworking unit.

Any combination of real subnetworks and relay systems may be referred to as an intermediate system.

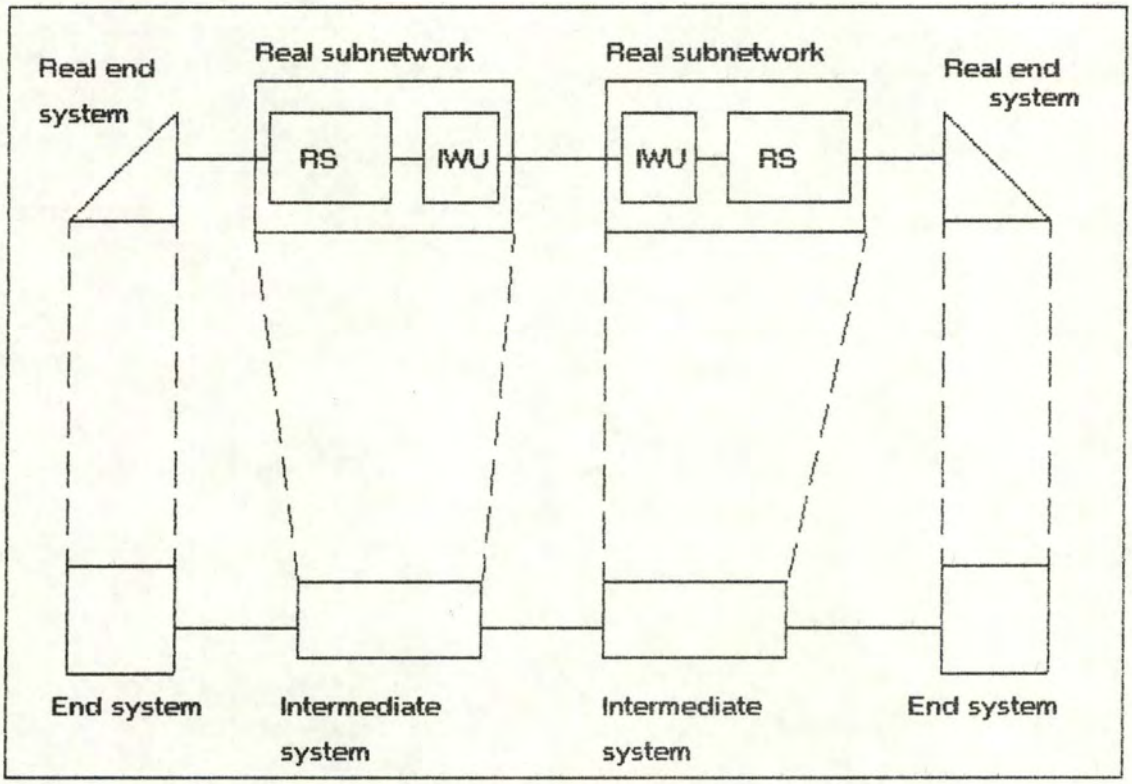


Figure 3.2: Interworking units.

Figure 3.2 shows a representation of each real subnetwork (RS) with an interworking unit (IWU) acting as an intermediate system.

Figure 3.3 shows the same picture for the real world but the abstraction is different because the real subnetworks alone are viewed as intermediate systems and all the interworking units together form an intermediate system.

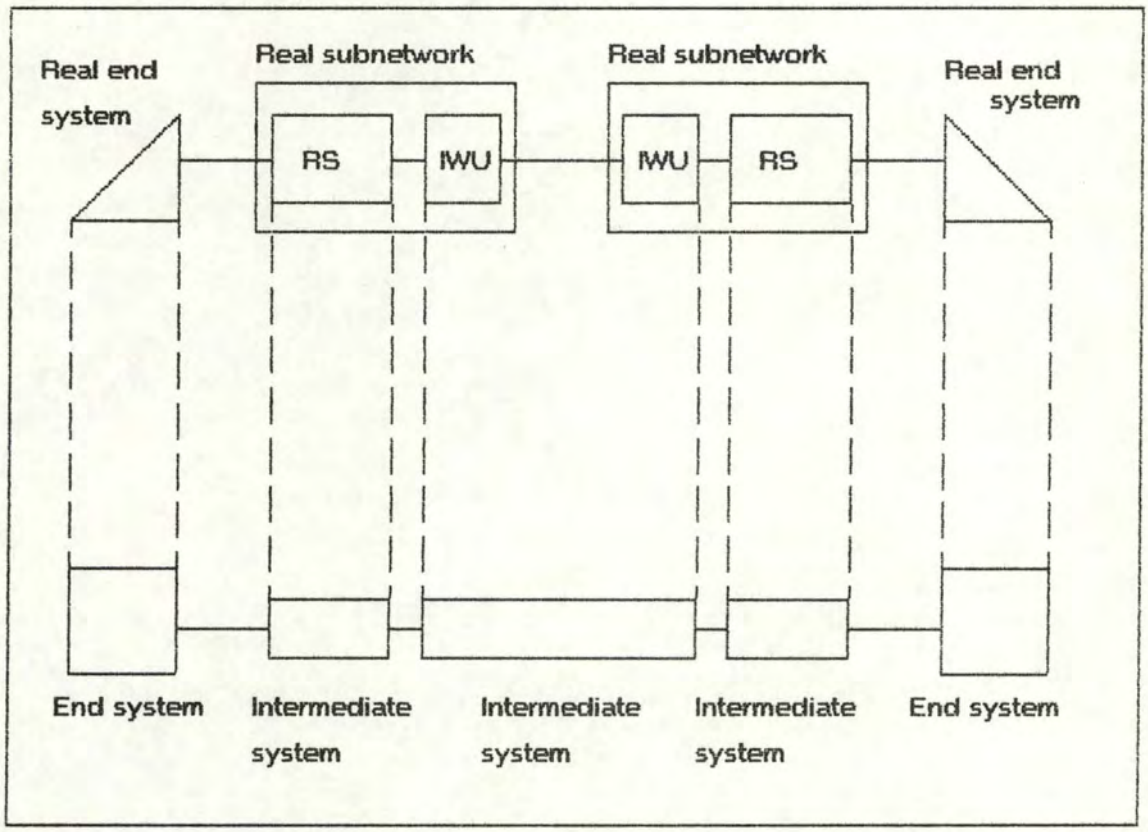


Figure 3.3: Interworking units and intermediate systems.

3.3.3 Real subnetworks and subnetworks

The abstract representation of a real subnetwork is a single subnetwork or an intermediate system.
The real subnetworks include:

1- the carrier supplied public networks
2- the privately supplied and used networks
3- the local area networks
4- the networks defined above with associated interfacing equipment and/or interworking unit(s), considered together as forming a real subnetwork.

3.3.4 Relay systems and interworking units

A relay system is an abstraction of a piece of equipment referred to in the real world as an interworking unit. The purpose of an interworking unit is to facilitate the interconnection of distinct real subnetworks.

3.4 Organization of the network layer

The presentation of the network layer is given in chapter 2. We will just give in figure 3.4 the graphical representation of the network layer to recall the different roles.

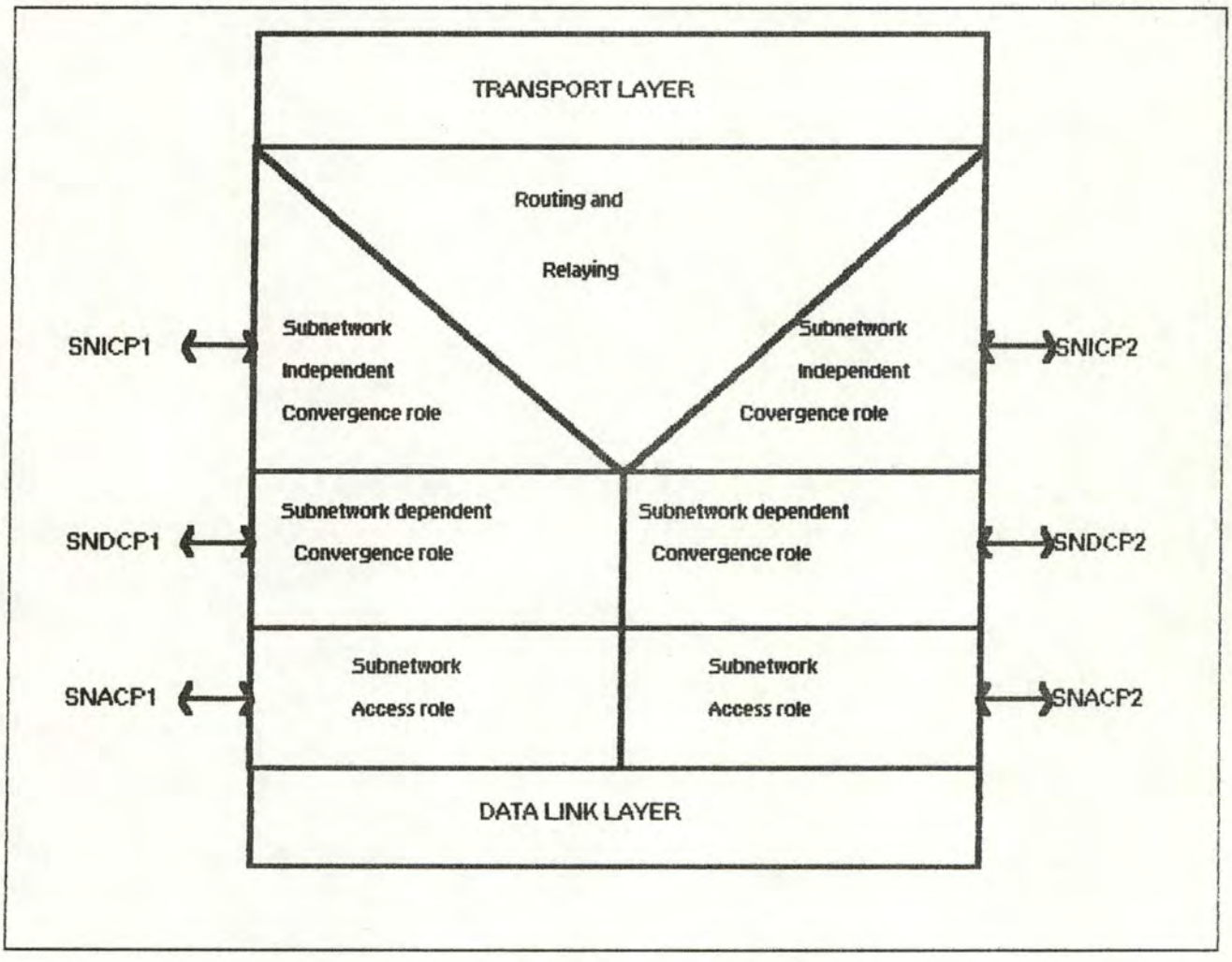


Figure 3.4: The organization of the network layer.

3.5 Different scenarios of interconnection

We will present in this chapter different scenarios of interconnection. We will not examine all of them. The reader can refer to [I8648] to find a complete description of the interconnection scenarios.

3.5.1 Single data-link/single subnetwork interconnection

Figure 3.5 shows the possibility for interconnecting over a single data-link and a single subnetwork. Two cases can be distinguished: the use of a single network layer protocol and the use of multiple network layer protocol. The left part of the picture illustrates the case for a single protocol and the right part ilustrates a multiple protocol. To really understand the different protocols of the network layer, the reader can refer to the explanation of the network sublayers in section 2.4.5.

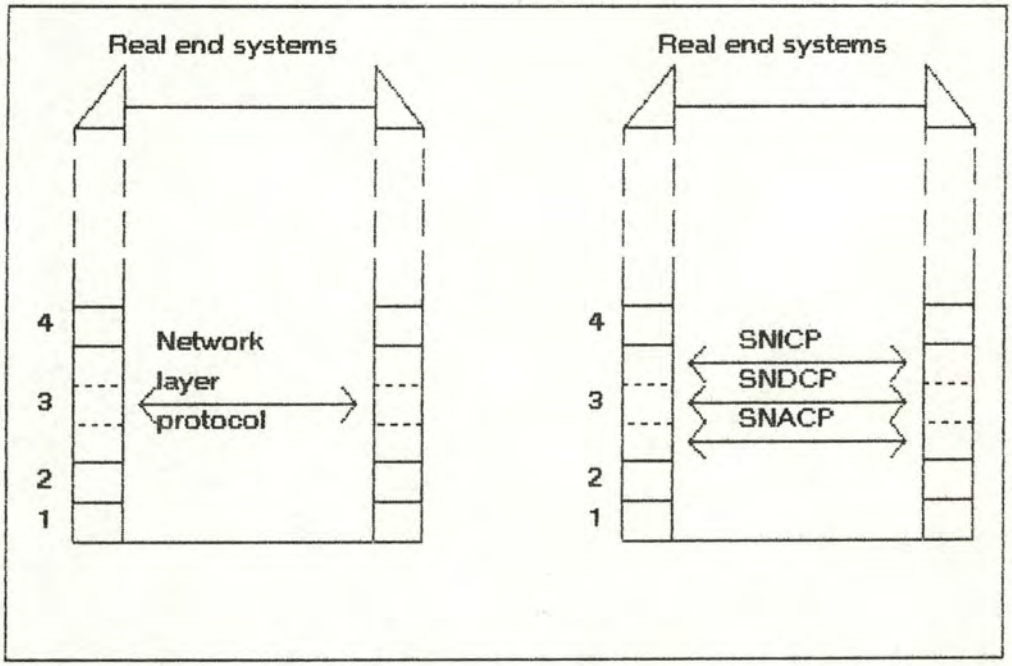


Figure 3.5: Single data-link/ single subnetwork

3.5.2 Subnetworks with all elements of the network service

Figure 3.6 illustrates the interconnection of subnetworks supporting all the elements of the OSI network service and using an interworking unit separated or not. The upper part of the picture illustrates the use of a single protocol and the lower part, the use of multiple protocols. In the picture RES means real end system, NLP means network layer protocol.

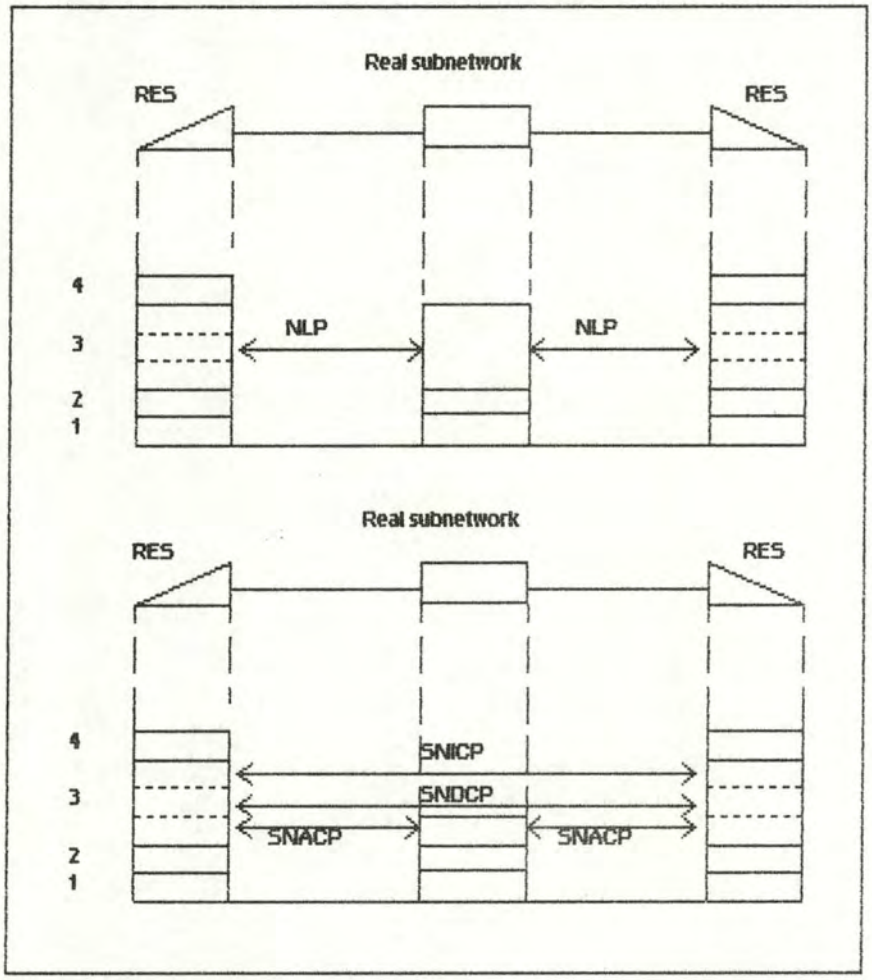


Figure 3.6: Interconnection of full-OSI subnetworks.

3.5.3 Interconnection involving internetwork protocol

Finally, we will show a scenario requiring the use of internetworking protocol Figure 3.7 shows the interconnection of two real subnetworks via a separate interworking unit. One of the subnetworks supports all the elements of the OSI network service, the other one does not.

In the picture, NLP means network layer protocol, IWU means interworking unit, RES means real end system, RS means real subnetwork.

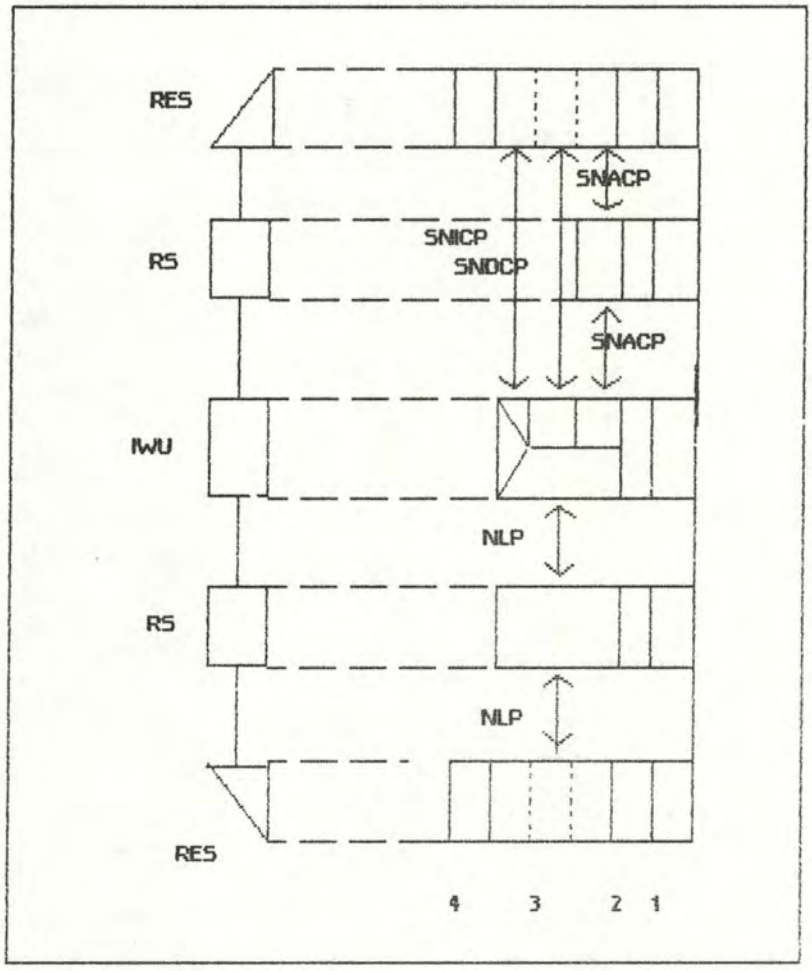


Figure 3.7: Scenario with internetwork protocol

These different scenarios show the complexity and the power of the interconnection in the OSI environment.

4. OSI network management

This chapter is the presentation of the network management for OSI. We will present the application field, the definitions and abbreviations related to the OSI management, the concept of management in the OSI point of view, the model developed by OSI for the network management, the specifics of OSI management, the management information base and the different services provided by OSI management.
The level of finition of the different documents about network management is various, e.g. the management framework ([I7498]) is under DIS form, the management specifics ([I0980] to [I0985]) are still under draft form. This was the main difficulty of this chapter to preserve a continuity in the text.

4.1 Application field

The abstract model developed in this chapter covers all activities that relate to the management of the OSI environment and applies to the definition and specification of services and protocols. These enable the planning, the organisation, the supervision and the control of the interconnection services that form part of an overall distributed information processing system. This model does not specify any particular implementation or interconnection technology.

4.2 Definitions and abbreviations

All definitions and abbreviations concern the international standard [I7498].

4.2.1 Definitions

OSI management: The facilities provided by the operation of systems management and layer management to supervise and control OSI resources.

OSI environment: The OSI resources which enable information processing systems to communicate openly, i.e. to conform to the services and protocols of open systems interconnection.

Local system environment: The resources which exist in a real open system, but which are outside the OSI environment.

Management information base: A conceptual composite of information of an OSI management nature, within an open system.

Systems management application process: The set of OSI systems management functions on an open system.

4.2.2 Abbreviations

MIB: Management information base.

SMAE: Systems management application entity.

SMAP: Systems management application process.

4.3 The concept of OSI management

The application field and the definitions having been presented, we will explain now the conceptual view of network management for OSI. This is done through a presentation of the environment of the network management, the requirement of the users, the resources involved in management, the facilities that could be provided to the people wanting network management.
All these concepts are developed in the working draft about the network management framework [I0391]. Some parts were already developed in detail by OSI, they will appear under the same form in the section, others were still under draft form, we have tried to make them readable.

4.3.1 The environment

The OSI management environment is the subset of the total OSI environment concerned with the tools and services needed to control and supervise interconnection activities and OSI resources. From a manager point of view, the OSI management environment includes the capability to gather data and exercise control and the capability to maintain an awareness of the status of OSI

resources. Some human responsibilities can be delegated to an automated process, but human beings are ultimately responsible for managing the OSI environment. This delegation can be translated in terms of self management of the open system and in terms of co-operation with other open system, through the exchange of information, to perform co-ordinated management activities.

4.3.2 The user's requirements

OSI has identified the requirements of the users for an OSI management system. The managers need activities which enable them to plan, organise, supervise, control and account for the use of interconnection services. This means facilities for a long term management of the open systems. They need facilities to ensure predictable communications behaviour and facilities which provide for information protection and for the authentification of sources and destinations for transmitted data. This means facilities for monitoring the open systems.

4.3.3 The OSI resources

The OSI resources include the facilities needed to operate the OSI protocols and the management data which provide information relating to the status of the OSI management environment.

4.3.4 The management facilities

We will give a short description of the OSI management facilities. These will be extended in terms of services in section 4.7. The requirements for OSI management are supported by a number of management services. Each service provides a range of facilities which may be executed either by local operation or by communication of information between open systems, or both. We can distinguish the following services: fault management, accounting management, configuration and name management, performance management, security management. The list of examples given for each service is not exhaustive.

A. The fault management service:

The fault management service is the set of facilities allowing the detection, the isolation and the correction of abnormal operations in the OSI environment. This service provides facilities to:

-Maintain and examine error logs.
-Accept and notify error detection.
-Trace faults.
-Carry out sequences of diagnostic tests.
-Correct faults.

B. The accounting management service:

The accounting management service is the set of facilities which allows the setting of charges for the use of OSI resources and the identification of the using cost of OSI resources. This service provides facilities to:

-Inform users of cost incurred or resources consumed.
-Enable accounting limits for use of OSI resources to be set.
-Enable costs to be combined where multiple OSI resources are invoked to achieve a given communication objective.

C. The configuration and name management service:

The configuration and name management service are the set of facilities which permits to control OSI resources by identifying them and collecting or providing them data. The aim of this service is to provide assistance for continuous operation of the interconnection services. This service provides facilities to:

-Initialize and close down OSI resources.
-Change the open system configuration.
-Set the open system parameters.

D. The performance management service:

The performance management service provides facilities allowing the evaluation of OSI resources behaviour and the effectiveness of communication activities. This service provides facilities to:

-Maintain and examine logs of system state histories.
-Gather data giving information on the OSI resources.

E. The security management service:

The security management service provides facilities for the protection of OSI resources. This means facilities to manage:

-Acces control.
-Encryption and key.
-Authentication.

4.4 The model for OSI management

OSI has developed a model to adapt its conceptual view to a possible implementation. We will present the structure of the management through different levels of management, we will give a short presentation of the management information base which will be developed later and finally, we will give the model developed by OSI for network management.

4.4.1 Overview

OSI management consists of two sets of activities needed to control, coordinate and monitor the conditions which allow communications to take place in the OSI environment:

-activities related to the interworking of OSI management components across open systems. This includes the means by which:

[] the management information is distributed between open systems.

[] the management information is accessed by remote open systems.

[] the execution of management processing is controlled.

-activities related to the management needs of individual open system for both layer specific and overall system. This includes the means by which:

[] the management information is stored and retrieved.

[] the management information is distributed among the layers.

The model presented in the next subsections is described both from the point of view of the flow of control amongst processes and from the point of view of the corresponding flow of information amongst entities.

4.4.2 The management structure

Three distinct categories of OSI management can be distinguished: systems management, (N)-layer management and (N)-layer operation. (N.B. Previously, the application management was inside the scope of OSI management, but it has now been separated .)

Systems management: relates to management of OSI resources and their status across all layers of the OSI architecture. It is the only means by which OSI management of multiple layers is assured. The systems management communication is established by the application-layer protocol. The systems management functions on an open system are collectively known as the systems management application-process (SMAP). The aspect of the SMAP which is involved in OSI communication is the system management application-entity (SMAE). The system management is an horizontal and vertical management: horizontal because it involves all the open systems which must be managed and vertical because it involves all the layers of these systems.
The following not-exhaustive list is typical for activities which fall into this category.

- activation and desactivation management which include:

  []activation, maintenance and termination of OSI resources distributed in open systems.

  []program loading functions.

[]establishment, maintenance and release of connections between management entities.

[]operational parameter initialization and modification.

- monitoring which includes:

[]reporting status or status changes.

[]reporting statistics.

- error control which includes:

[]error detection and diagnostics functions.

[]reconfiguration and restart.

Figure 4.1 shows the notion of OSI systems management. A SMAP communicates with an SMAE to ask management services. These services are provided by the SMAE through the communication with other SMAE.

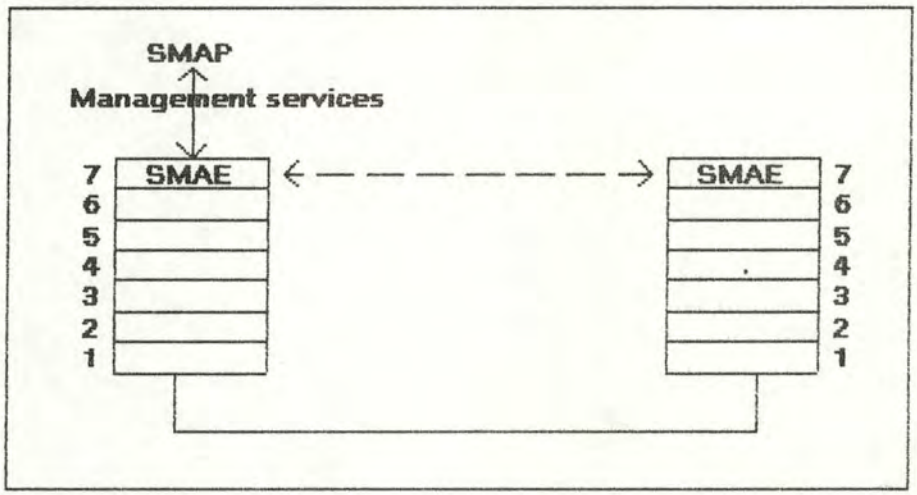


Figure 4.1: OSI systems management

(N)-layer management: provides mechanisms for the monitoring, control and coordination of the (N)-layer OSI resources used to perform communication activities within an (N)-layer. (N)-layer management entities communicate with each other to support management control of (N)-layer OSI resources used for communication between their respective open systems. The (N)-layer management can affect multiple instances of communications.

The (N)-layer management communication is made through systems management protocols or through (N)-layer management protocols, or both.
The (N)layer-management is an horizontal management inside one layer. Figure 4.2 shows an (N)-layer with (N)-layer management entities **"(N)-LM"** managing the OSI resources of three instances of communication between (N)-layer entities **"(N)-ENT"**.

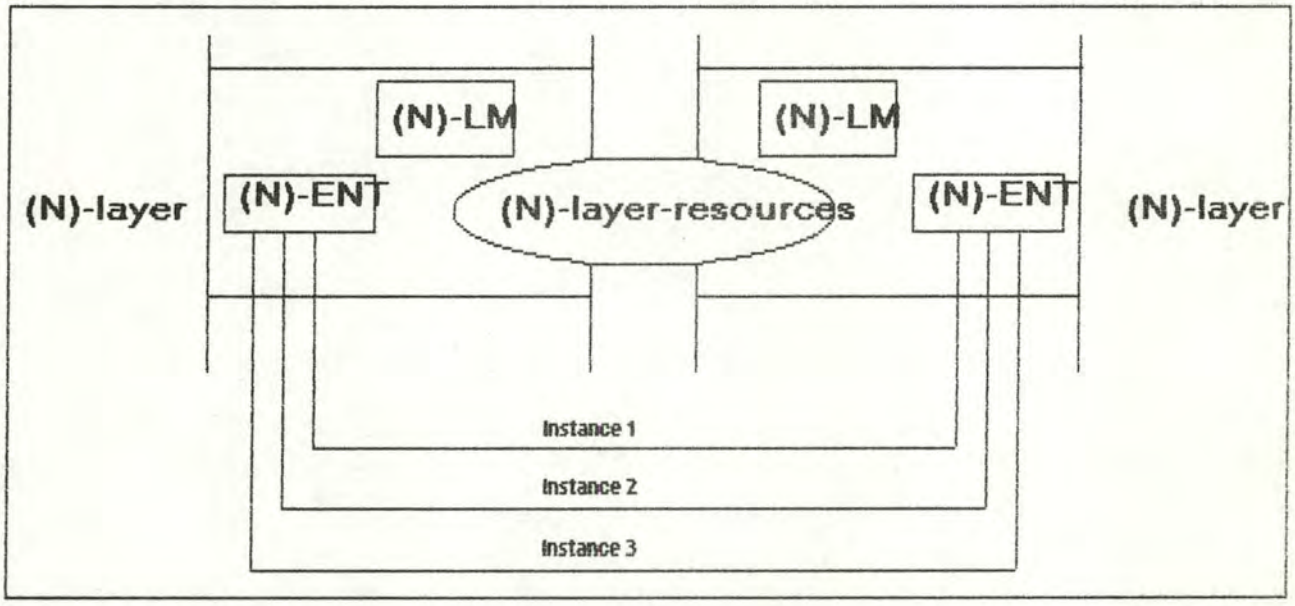


Figure 4.2: The OSI (N)-layer management

(N)-layer operation: provides the set of facilities which controls and manages a single instance of communication. It is only horizontal management (on ONE layer) with a SINGLE instance of communication.

4.4.3 The management information base

The management information base will be shown in detail in section 4.6. But to present the model of OSI management, a short description appears as necessary.
A management information base is the set of OSI Management data in an open system available to the OSI environment. These data are communicated between open systems using an OSI management protocol (not defined yet by OSI).

The concept of MIB does not imply any particular implementation for the storage or for the form of the data. These considerations are outside the scope of OSI and are a matter of local concern. But the data within the MIB are structured according to the requirements of the OSI management processes which need to access it.

4.4.4 The model

From OSI point of view, the management is modelled as being effected through a set of management processes. These processes may be distributed over a number of open systems or located on a single system. The communication between management processes is done by protocols (not yet) defined by OSI. We will distinguish two types of information: the control information and the management information.

A. The control information: are information for the control of the resources.
The management processes receive control information:

-from people and/or software acting as administrative agents local to a management process.
-from remote systems from their SMAEs, their (N)-layer management entities or their (N)-entities.

The management processes control:

-directly the OSI resources located in the same open system.
-the OSI resources located on other open systems by protocol exchange through their SMAEs, their (N)-layer management entities or their (N)-entities.

B. The management information: All the OSI management information are part of the OSI management information base. These information are provided by:

- local management agents.
- remote open systems through their:
  - [] systems management protocols.

[] (N)-layer management protocols.

[] (N)-layer protocols.

Figure 4.3 is a graphical representation of the OSI model for management. A system management process (SMP) including the system management application entities (SMAP) forms with the management information base (MIB) the local environment. The open system environment includes the (N)-layer managers, the (N)-layer entities and the system management application entity (SMAE) in layer 7.

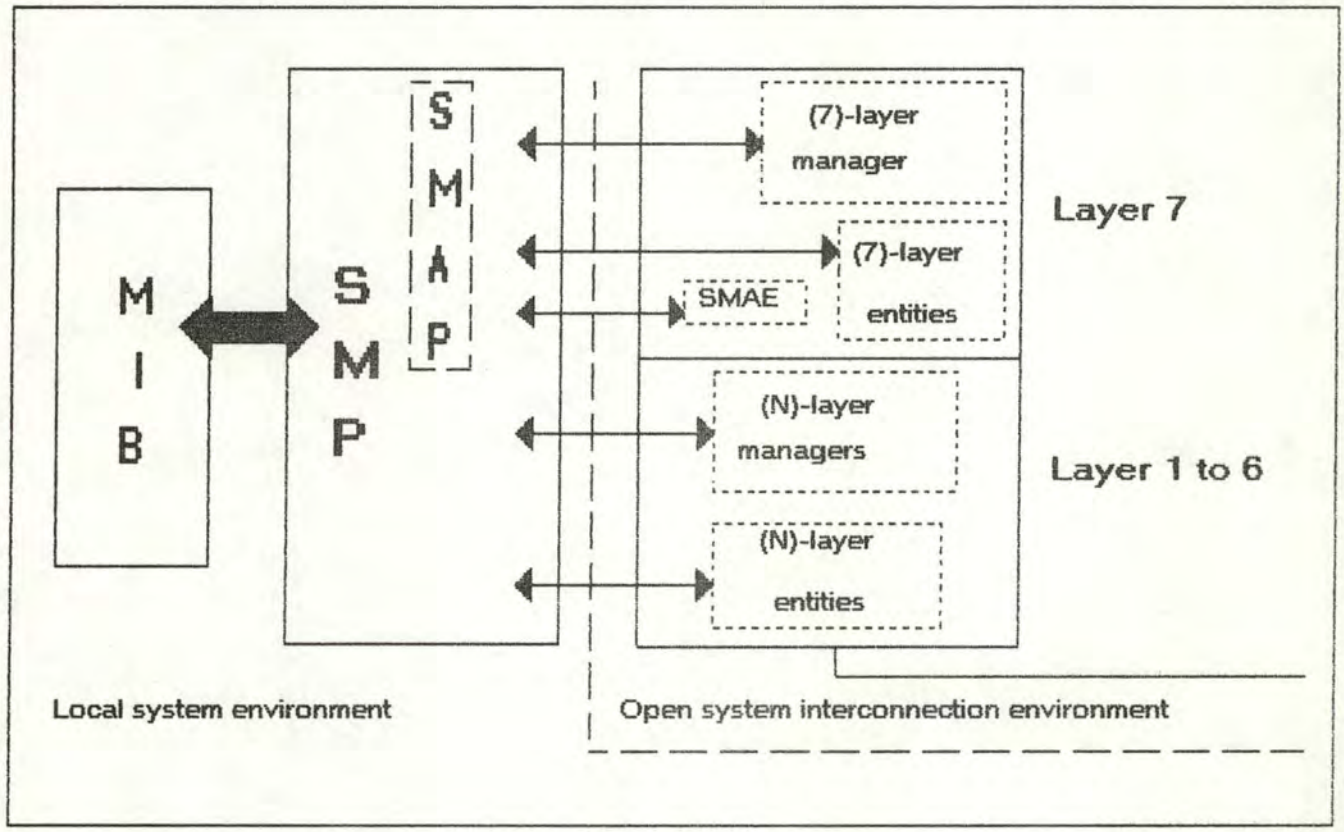


Figure 4.3: OSI management model

We will present the system management process in terms of boundaries to see what is in the field of OSI standardization or not .
The boundaries show where the SMP ends and where other objects (inside or outside the open system) begin.

Figure 4.4 shows the SMP with 4 boundaries.

A. Boundary 1 is the boundary between the SMP and the people and software that request services to the SMP. The services request passes through this boundary to invoke one or more management functions. It is local and is not subject to OSI management standards.

B. Boundary 2 is the boundary between the SMAP and the SMAE. The SMAE is an application entity within an open system that communicates via a system management protocol with other SMAEs in other open systems. This is in the field of OSI by the definition of protocols.

C. Boundary 3 is the boundary between the SMP and the individual layer managers. Data and control information pass through this boundary. This boundary provides for each layer manager, a way to gain access to external parameters. This is in the field of OSI by the definition of protocols.

D. Boundary 4 is the boundary between the SMP and the individual layer entities. Data information pass through this boundary. This is in the field of OSI by the definition of protocols.

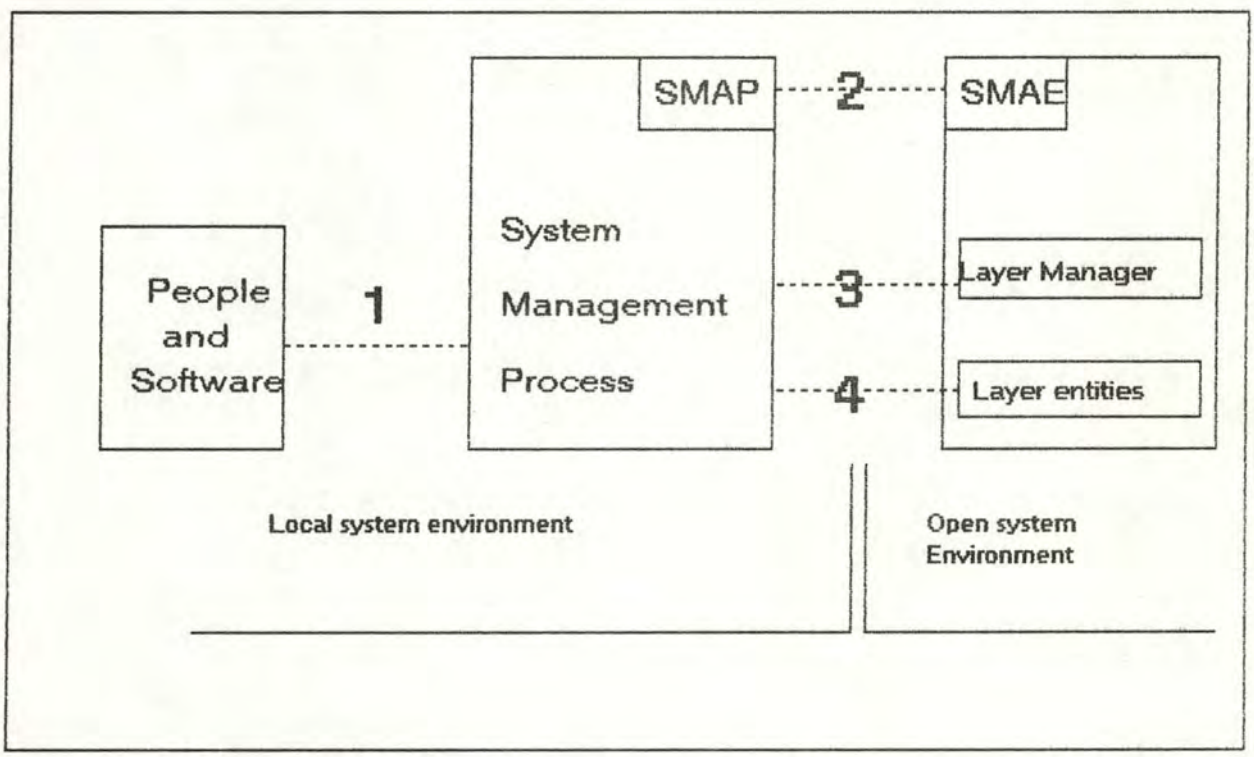


Figure 4.4: Boundaries of the SMP

4.5 The OSI management specifics

We presented the conceptual approach and the model of network management. We will now give the specifics of the OSI management by presenting the organization of the different levels of management and the area of standardization.

4.5.1 The organization of the systems management

The normal method for exchanging OSI management information is to use the systems management communication between SMAEs. Systems management protocols are application-protocols. When the system management processes communicate with each other through their SMAEs, they use the services provided by the lower six levels of ISO 7498.

All systems do not have the full functionality of the seven layer specified by ISO 7498. We can distinguish two situations:

-the initial source and the ultimate receiver for the transfer of management data are full ISO, but some relays do not have the full functionality. So in this case, there is no problem, because the "not-full" network just need to serve as a relay between the two "full" networks.

-the initial source or the ultimate receiver for the transfer of management data do not support the full functionality of the seven layer. But in this case the minimal requirement is the support of the system management protocol and a sufficient communication capability in the seven layer.

4.5.2 The organization of the (N)-layer management

The (N)-layer management supports the monitoring, control and coordination of the (N)-layer OSI resources. The (N)-layer management protocols are supported by protocols of the lower (N-1)-layers specified in ISO 7498. The (N)-layer management protocols cannot be used by the (N+1)-layer management

protocols. Figure 4.5 shows the relationships between the protocols of different layers. Each arrow shows a relation of use.

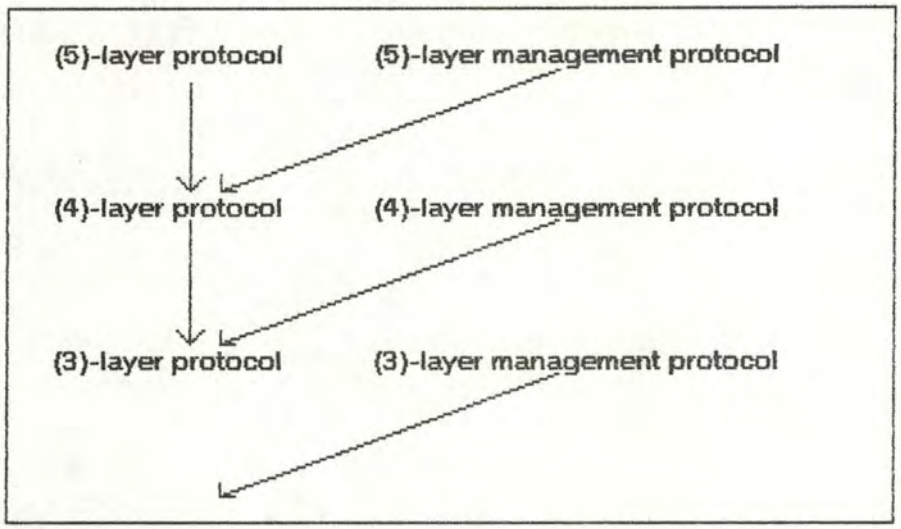


Figure 4.5: The (N)-layer management protocol.

The (N)-layer management protocol can only convey management information between peer (N)-layer entities pertinent to the (N)-subsystems in which these entities reside.

"The (N)-layer management protocol should only be used instead of systems management protocols either when systems management services may not be available to exchange layer management information or when the exchange is inhibited by the use of upper layer functions. [I7498]"

The functions of the (N)-layer management protocol are:

-the communication of the value of the parameters related to OSI resources of the (N)-layer.

-the testing of the functionality provided by the (N-1)-layer.

-the conveying of information about faults or diagnostics data related to the OSI resources of the (N)-layer.

4.5.3 The organization of the (N)-management-protocol

An (N)-management-protocol provides monitoring and control of a single instance of communication within the (N)-layer.
There exist management functions within the (N)-protocols in all seven layers.
The protocol is responsible for the distinction of the management information and of the normal information that it carries for other purpose.
The (N)-protocol carries management information such as:

1-parameters carried in connection establishment that apply to the specific instance of communication.

2-parameters carried in particular packet data units which can modify the environment in which this instance of communication operates.

3-error information describing faults encountered during the operation of that specific instance of communication.

4-parameters carried in connection release which report information pertaining to that specific instance of communication.

4.5.4 The OSI management standardization.

The areas of standardization are the abstract syntax and semantics of the information contained in a MIB and the services and protocols used to transfer management information between open systems.

4.6 The management information base

4.6.1 Introduction

We will now present the management information base. We will give the role, the contents, the access and the MIB users.

4.6.2 The role of the MIB

"The MIB is the conceptual repository in an open system for all information needed to operate in the sense of the reference model. [I0391]"
This concept does not suggest any form for the storage of this information and its implementation is a matter of local concern.
Information are to be standardized only once a need for it to be transferred between open systems has been established.

The MIB is under the responsability of the system management process (SMAP) which also provides the access interface.

4.6.3 The contents of the MIB

The MIB contains information about open systems resources that are of interest to open system management functions.
Open system resources are:

- -(N)-SAPs
- -(N)-entities
- -application-processes
- -other objects that participate in communications

The attributes of a resource are the items of information about that specific resource. Every attribute consists in an attribute identifier part and an attribute value part. The attribute identifier uniquely identifies an instance of an attribute of a resource. The identifier may be composed of multiple nested attributes.
For example: a simple counter which counts the number of data units transmitted on a connection over a period of time on a specific (N)-SAP would require an attribute composed of the following sub-attributes:

- connection end-point
- connection identification
- start time
- end time

A second type of management information that is important to OSI is exchange information that does not reside in the MIB, but that are relays information about the resources to and from external management systems. These information are called transients of a resource and are of two types: actions or event reports.
We will take an example of the contents of the MIB for a particular layer: the transport layer. [I0382]

1. Information on a transport entity:

number of Network-Connections "open"
number of N-Connections not "open" (request not satisfied)
number of Transport-Connections "open"
number of T-Connections not "open"
N-Connection open at present
T-Connection open at present
number of protocol errors
number of RESET indications (Network provider generated)
number of disconnect indication (Network provider generated)

2. Information on transport connection:

time of initialization
time of termination
bytes transmitted to network
bytes received by network
bytes transmitted to session
bytes received by session
number of reset indications (network provider initiated)
number of disconnect indications (network provider initiated)
number of data TPDUs transmitted
number of control TPDUs transmitted

The information can be classified as follows:

- Event type information:
  . Counter (Errors, Timeout,...)

. Thresholds

- Structured information:
  . Directory information base
    - Application layer information
    - Network layer information
- Attribute type information:
  . Parameters (eg window size)
  . etc.

4.6.4 The access to the MIB

The access rights to the MIB and the control of the MIB are by nature different but are always under the responsability of the corresponding SMAP even if the requester is in another system. This rule is logical because the SMAP is the only one that knows the internal implementation of its corresponding MIB.

Two types of accesses are defined:

Access type 1: is a Read Only access performed for example by

- application processes
- system management processes

Access type 2: is Read/Write access restricted to a certain number of management functions with the need for controling data integrity and security. Examples of functions that can manage this access:

- error logging
- accounting
- directory service

Figure 4.6 shows the different accesses to the MIB.

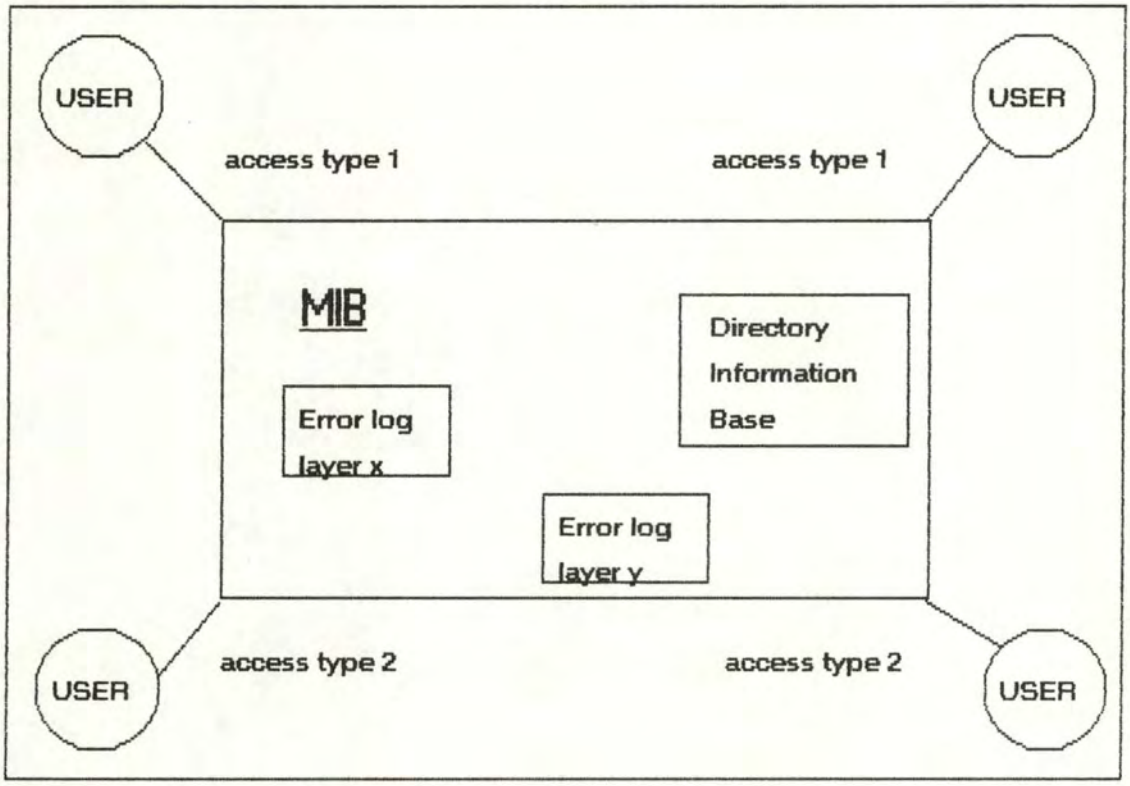


Figure 4.6: Access types to the MIB

4.6.5 The MIB users

The MIB users can also be classified.

-Open system application processes which access the MIB through a generalized service such as the directory service, or directly.

-System management processes which includes OSI management application process and (N)-layer managers.

Figure 4.7 shows the accesses of the different MIB users.

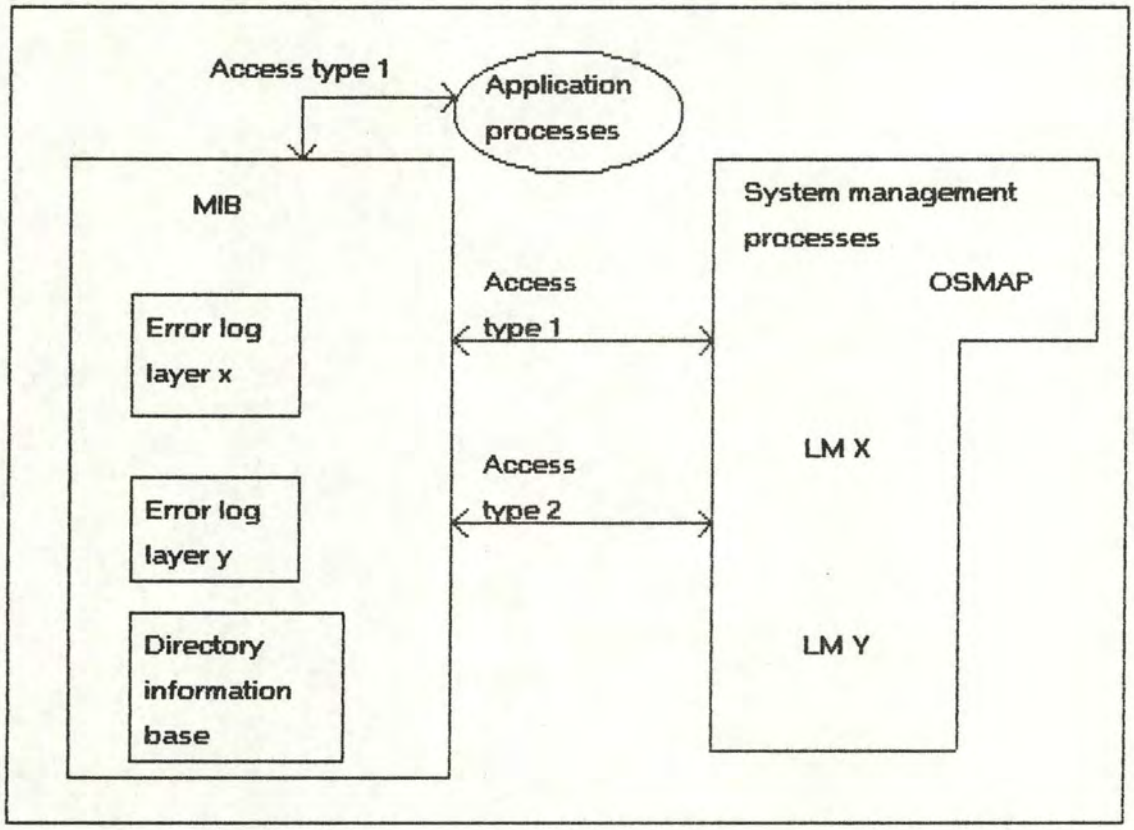


Figure 4.7: MIB users.

4.7 The management information services

4.7.1 Introduction

We will present the definition of the different services provided for the management of a system. It is the only part that is nearly fully defined by OSI (under draft forms).

The services are divided into two parts: the common management information service and the specific services such as fault management, security management,...

For all the services presented below, we will follow the structure developed by OSI in the draft quoted in the beginning of each section.

4.7.2 Common management information service

A. Overview

The common management information service element (CMIS) provides a generalized set of services for the transfer of management information and control. Three categories are identified for the transfer of management information. These are control activities, information exchange activities and event notification. These three categories will be defined in terms of facilities.

B. Facilities

Six main services provide the three categories of management for the CMIS. These management information service elements are used by open systems to communicate with each other to provide the management functions.

1- Event notification:

[] **Event report**: is a service element used by a SMAE to report some unsolicitated events of a resource to another SMAE in another open system.

[] **Confirmed event report**: is the service element used by a SMAE to report some unsolicitated events of a resource to a SMAE in another open system, from which it expects a response.

2- Information transfer:

[] **Get**: is the service element used by a SMAE to request transfer of management information from another SMAE in another open system.

3- Control:

[] **Set**: this facility provides the capability to request a SMAE in

another open system to set the values of attributes within the open system. The SET is the primary mechanism for resource control.

[] **Action:** this facility provides the general capability to request a SMAE in another open system to perform an operation. This facility is used only when operations cannot be made via the attribute value manipulation capabilities of set.

[] **Compare**: this facility provides the capability to request that a SMAE in another open system compares the value of attributes within the open system to a specific value and return the result.

[] **Blocking**: This facility provides the capability to combine the other services. Using blocking, a SMAE will cause several operations to be performed on resources situated in another SMAE in another open system.

C: Procedures:

For the services elements, the procedures have already been defined by OSI. These procedures consist of four primitives: M-service-REQ, M-service-IND, M-service-RSP and M-service-CONF.
M-service-REQ is the primitive used by the initiator SMAE the REQuest a service to its responder SMAE.
The service provider issues an M-service-IND primitive to indicate the service requested to the responder SMAE.
The responder issues an M-service-RSP primitive to report the acceptance or the reject of the request to the initiator. This primitive is used only when a response is requested.
The service provider issues an M-service-CONF primitive to confirm to the initiator SMAE.
The parameters for these primitives are defined in [I1373]. These parameters will be for example: the identificator of the resource, the status,...
Figure 4.8 shows the sequence of primitives for the ACTION service. Two SMAEs, located on two different open systems, are exchanging

management services. The SMAE on open system 1 request an ACTION on the SMAE of open system 2. To perform that action, it sends a M-ACTION-REQ and waits for an M-ACTION-CONF. The SMAE on open system 2 receives an M-ACTION-IND, it performs the action or not, and sends the results by an M-ACTION-RSP.

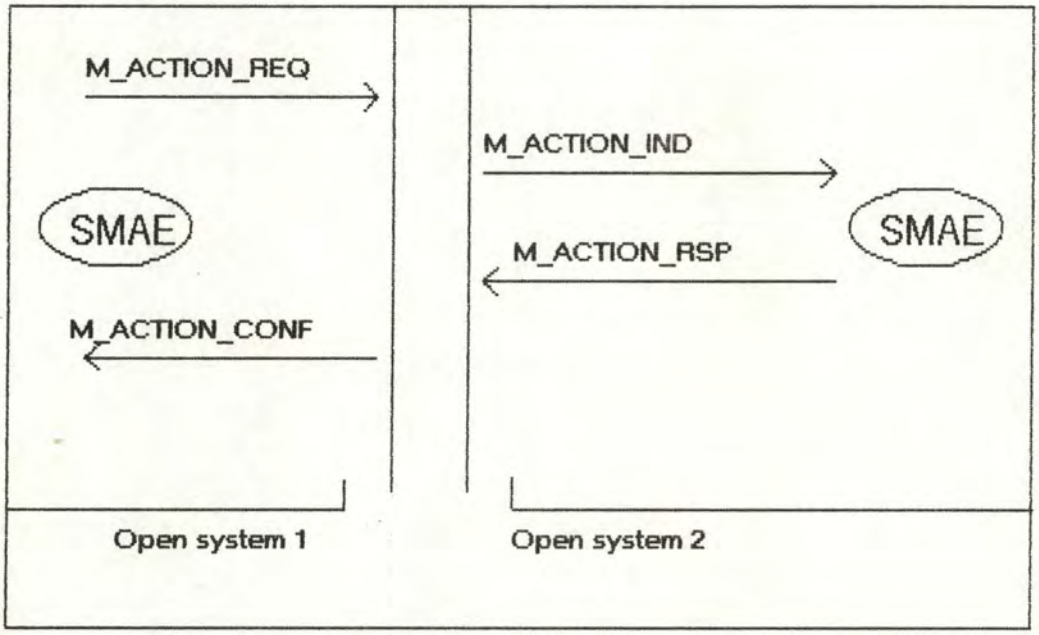


Figure 4.8: CMIS procedure.

4.7.3 Fault management information service

A. Overview

"Fault management is used by the system administration to assist in making decisions with regard abnormal operation of an open system and to direct the corrective action on the faulty resources. [I0984]"

B. Facilities

[] Spontaneous error reporting facility: is used by an agent (SMAP) to send error reports to a manager (SMAP). For this purpose, it uses the CMIS facility of report sending: M-CONFIRMED-EVENT-REPORT.

[] **Cumulative error gathering facility**: enables a manager (SMAP) to periodically request error counter information from an agent (SMAP) using the CMIS M-GET-ATTRIBUTES service element.

[] **Error threshold alarm reporting facility**: is used to send threshold report from an agent (SMAP) to a manager (SMAP) using the CMIS M-CONFIRMED-EVENT-REPORT service element.

[] **Continuous monitoring facility**: enables an agent (SMAP) to send all event reports to a manager (SMAP) using the CMIS M-CONFIRMED-EVENT-REPORT service element.

[] **Confidence testing facility**: allows a manager (SMAP) to direct an agent (SMAP) to perform a test on a resource to determine if it is capable of performing its service. It uses the CMIS M-ACTION service element

[] **Diagnostic testing facility**: allows a manager (SMAP) to direct an agent (SMAP) to perform a test on a resource to assist the diagnosis of a fault. It uses the CMIS M-ACTION service element.

[] **Communication path tracing facility**: provides to a SMAP the way to initiate a path trace to a series of other SMAPs to check the underlying communication facility.

[] **Trouble ticketing facility**: provides a mechanism to one SMAP to report activities on a particular problem to other SMAPs so that they may cooperate to the total fault management of the OSI resources.

C: Concepts and models of fault management

Fault management concepts: fault management is the management of the abnormal operation of the system. Faults are those things that cause the system to operate in an abnormal way. The origin of the faults can be internal (e.g. a broken component) or external (e.g. environmental influence). Faults appear as errors in the operation of the system. Once an error is detected, signalling the existence of a fault, the cause and/or the location of the fault must be found. Finally, corrective actions can be

taken so that the system can perform its intented function. Three concepts can so be identified: the fault detection, the fault diagnosis and the fault correction.

A-The fault detection: Three different services can be provided for the fault detection: a resource can detect a fault during its normal operation and can generate an error report ; the running confidence tests can detect faults due to uncovered resource ; and fault conditions can be anticipated by thresholds being exceeded.

B-The fault diagnosis: Once a fault is detected, it is often important to determine the exact location of the fault so that it is possible to take repair actions. This can be done by analyzing symptoms (past events in a log file) or by executing diagnostics tests.

C- The fault correction: When a non tolerable fault is detected and located in a repairable resource element, repair actions can be taken to return the resource to a serviceable state. These repair actions can be hardware or software.

Model of fault management: "Fault management can be a distributed application, distributed across multiple open systems, each system containing one or more application processes each called a System Management Application Process. [I0984]"

The SMAP can :

a. receive error reports and other management information from the resources in its own open system.

b. produce reports on detected errors, running of tests and so on, that may be sent to other SMAPs residing in remote open systems.

c. control the management actions on a remote open system.

d. notify the error status and other management information to the system administration of its open system.

Only items b and c are in the field of standardization of ISO.

4.7.4 Accounting management information service

A. Overview

"The aim of the configuration management is to permit the user to manage the costs incurred arising from charges made for the use of communication resources. [I0981]" The charges can be classified as: standing (fixed) charges which are independent of the use made of the communications resources and are imposed to recover costs of a service whether used or not; usage charges are imposed to recover costs of exercising a particular instance of communication.

The standing charge is known by the user. The usage charge is determined by different parameters such as: the time of day at wich communication instance occurs, the duration of the dedication of the resources, the quality of the service required by the subscriber, the added-value services used,...

Two classes of services are identified by OSI for the accounting management: the network layer accounting class and the application layer accounting class. The network layer accounting class collects the information based on the initiator of the information and the receiver of the information. Nothing else is defined yet on the concept of classes or for the application layer accounting class.

B. Services

Four services are until now identified by OSI for the accounting management:

a. Collecting an accounting information:
An accounting node starts to collect an accounting information when a network-connect-request is invoked by the end system. It continues to collect information as long as the connection is active and stops when the connection -release is invoked. These information will be stored and transferred to the accounting process node.

b. Reporting an accounting information to end systems:
The end-user will report to different end systems following the

specification of the connection: he will report to the initiator only, or to the receiver only, or to both initiator and receiver, or to another end-user.

c. Reporting an accounting information to the accounting process node:
Before the accounting node's storage overflows, it informs an accounting information to an accounting process node which maintains an accounting information for billing.

d. Reading accounting information requested by an end-user:
After releasing a network connection, the end-user can inquire an accounting information from an accounting node via an accounting process node.

4.7.5 Configuration management information service

Only a few things are already defined by OSI for configuration management.
The configuration management services handle with configuration data. For OSI, " a configuration data is any information about OSI resources that are needed to manage the open system or the open system network. [I0982]"
The service provided by the configuration management is the definition, the collection, the monitoring, the management and the use of the configuration data. The configuration management includes the set of functions, data, messages and parameters used to:

- collect information about the system
- control the system state
- store the system state and state histories
- present the system state.

4.7.6 Perfomance management information service

Only a few things are already defined by OSI for the performance management.

"Performance management is the long term evaluation of the system. This management function requires gathering of statistical data in order to analyze trends in the operation of the communication between open systems. [I0983]"

The performance management includes a set of functions, data, messages and parameters used to:

- collect the system statistics
- control the collection of system statistics
- store the system statistics and historic of statistics
- analyze the system statistics
- present the system statistics.

4.7.7 Security management information service

The model for the security management is not yet defined by OSI.

The only information available is that "security management information is used by the system administration to assist in making management decisions with regard to the security of an open system. [I0980]"

This will include functions for the authentification management (distribution of passwords to entities), access control management, key management, security audit trails and event handling and MIB access control.

4.8 Evaluation of the model

This presentation leads us to formulate some conclusions about the OSI management model.

The structure proposed for management is interesting but it forgets perhaps the evaluation of the different classes of people and software that will receive the management information. The identification of users classes should be interesting.

The protocols that will be defined are interesting even for people that do not want to follow the OSI model because the trend seems (in first drafts) to be very wide, giving so the opportunity to use them in other systems than full-OSI systems.

The definition of the services is very wide too. It seems to cover all the domains requiring management. But to be certain, we must wait for the definition of the last services.

The complete model is difficult to implement for networking environment that is very heterogeneous.

At the disadvantage of the seven layer model: the redundancy inside the message will be aggravated here. Each layer manager will add information in the message.

So in conclusion, we can say that the management model is, at the moment, a good reference model, but it is too young to be implemented.

5. The CERN application: the COMS project.

The aim of this chapter is to present the multi-network management system which is developed by the European Council for Nuclear Research jointly with Digital Equipment Corporation: the COMS project (Common Operational Management System). To understand the need for this system, the networking environment will be first presented; this includes the different networks used in CERN and the gateways avalaible between these networks. After that, the concept of network management in a mono- and multi-network environment will be studied, and finally we will present the network management system and the COMS project.

5.1. The CERN networking environment

The CERN networking environment is very complex; twelve main networks can be identified . Figure 5.1 gives a presentation of these networks. The boxes represent the type of connection: "WORKS ON" means that a network uses the services provided by another one; "INTERCONNECTS VIA" means that the network interconnects two or many nets transparently via a third one. To show the complexity of the CERN networking environment, we will develop in detail the subsystem surrounded with a dotted line. EXCITE and DECnet will be developed too due to their importance for the CERN. And finally we will present briefly SNA, INDEX, EARN and UUCP. This forms three different logical subsets of networks that can be seen as disconnected for clearness reasons.

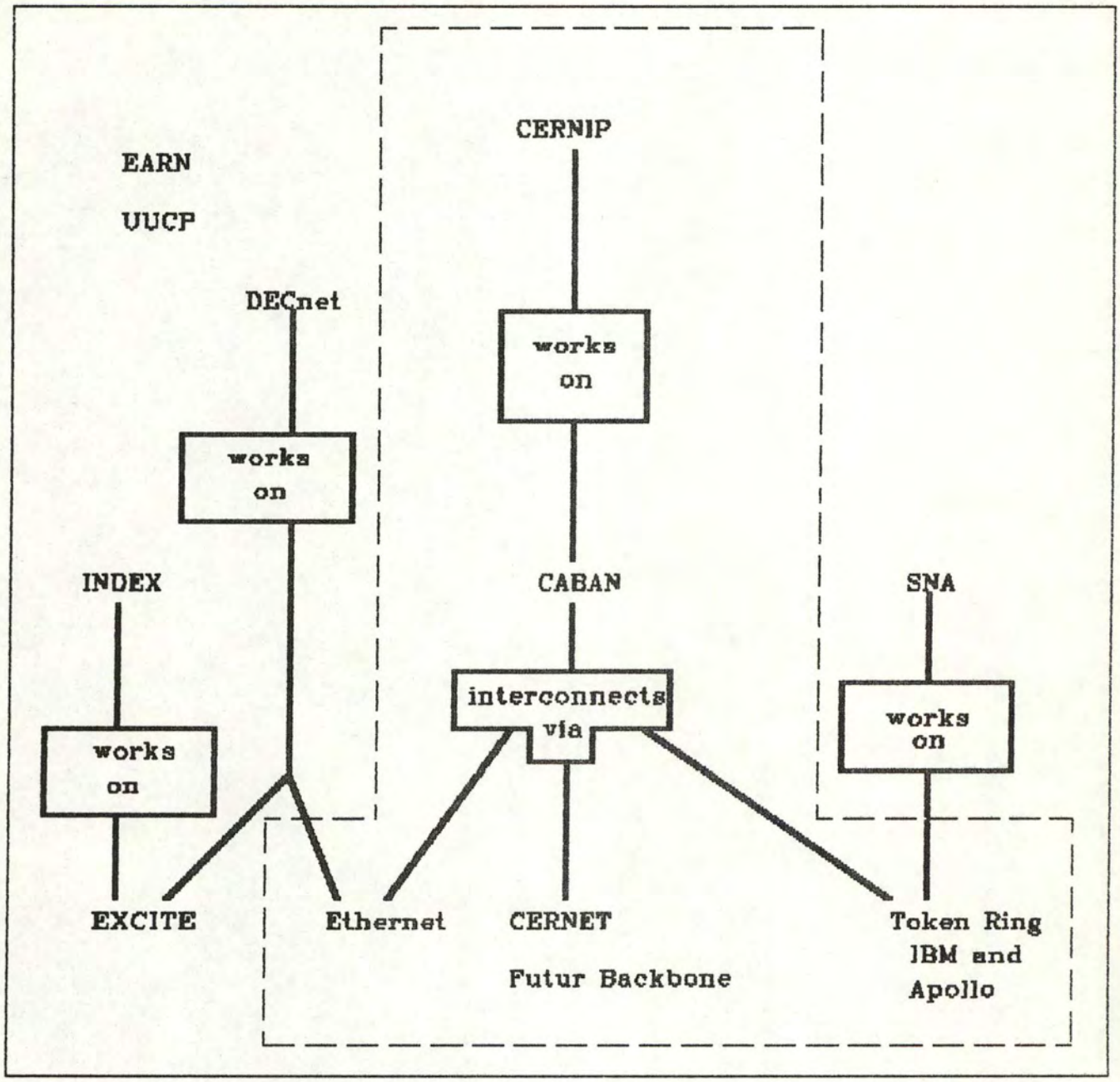


Figure 5.1: The CERN networking environment.

5.1.1 A first logical subset of networks

To be clear, we will take a bottom-up approach; this means: we will present first the networks implementing at least the physical level: IBM token ring, Apollo Domain token ring, Cernet and the futur backbone. Ethernet is included in this list even if it will not be developed. The reader can consult [GRE83] for more information about Ethernet. We will then make a synthesis of these networks showing the place they have in the OSI model. Then, we will present the CABAN network which implements MAC level for the interconnection of the networks of the first list. And finally, we will present CERNIP which implements the level 3 and 4 over the CABAN network.

For each network presented in depth, we will respectively present its logical architecture, its physical architecture, its architecture in CERN and its management requirement.

Let's imagine a small scenario to illustrate a typical situation in the CERN environment. "MessrsAndre, Bertrand, Claude and Denis are working on the same project. They all have a PC in their office and they would like to communicate with each others. In the office of Andre and Bertand, a Ethernet cable is passing , in the office of Claude there is a IBM token ring cable and in the office of Denis there is an Apollo Domain token ring cable."

We will show how the different networks are used to provide a way of communication between all these persons.

5.1.1.1 IBM token ring

IBM token ring is an implementation developed by IBM of the ISO 805/2 local area network standard. It is aimed at a multivendor environment with chips set and interfaces announced for most popular machines. It is used in CERN for the new accelerator LEP for the control of the equipments.

A. Architecture of the network

IBM token ring is a local area network developed by IBM. It uses a ring architecture on which a token is turning. The stations can speak when they have the token.

IBM token ring operates at eight megabits per second over a special wiring supplied by IBM.

Actually, in the CERN, there will be a mixture of media: twisted pair cables , TDM channels which operate on coaxial cable or glass fibers, glass fibers. Figure 5.2 shows a typical configuration of IBM token ring. Personal computers (PC) and personal workstations (PW) are connected to the ring as well as disk servers (DS) and gateways.

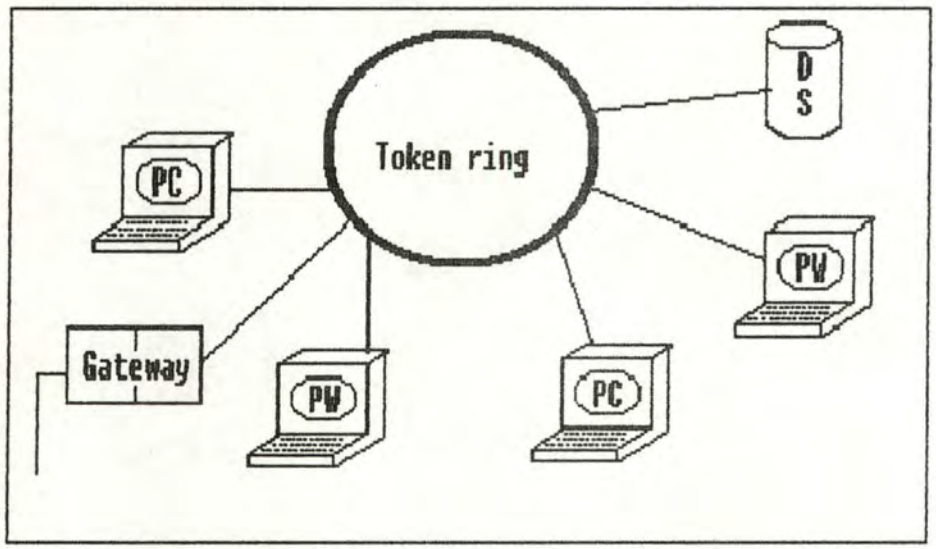


Figure 5.2: IBM token ring

B. Management requirements for this network

The management requirements for the IBM token ring are the configuration management and the monitoring of the network.

5.1.1.2 APOLLO Domain

Apollo Domain is a vendor specific network for networking Apollo workstations. [PET86] It has been developed to support Apollo distributed file system, Apollo interprocess communication and remote paging of the operating system. It is used in CERN by physicists for the development and the use of physics analysis programs.

A. Architecture of the network

The Apollo Domain is based on token ring type local area network to which all stations are connected.
The physical ring is a coaxial cable and allows a maximum bandwidth of twelve Megahertz.
More than one ring can be interconnected via an "internetwork media adaptor" which runs over Ethernets or other lines.
There are actually 35 nodes connected on two Domain rings.
Figure 5.3 shows the interconnection of two rings via a G700 line.
IMA is the internetwork media adaptor and PW is a personal workstation.

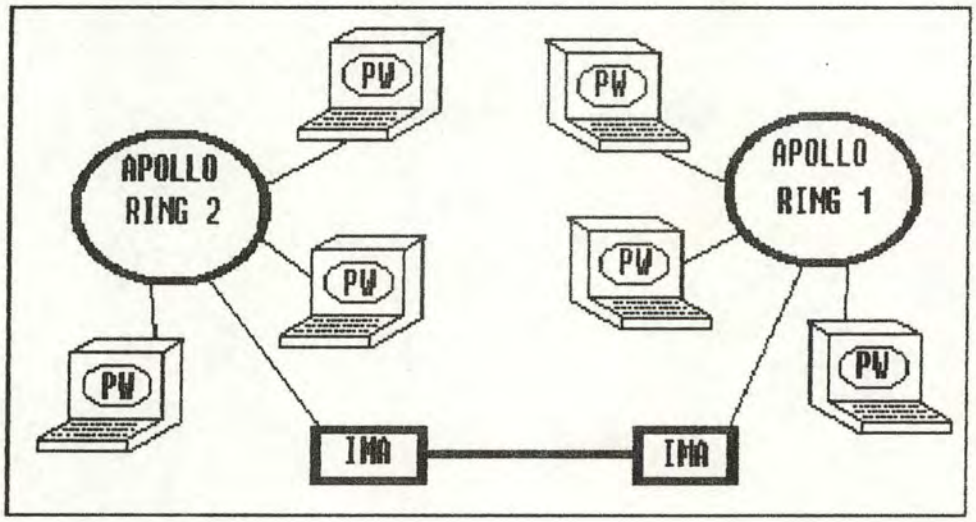


Figure 5.3 Apollo Domain.

B. Management requirements for this network

There are some existing tools for monitoring and managing the network, these tools should be integrated in the COMS project.

5.1.1.3 Cernet

A. Architecture of the network

In the mid-1970s, the CERN decided to build a general purpose network to be used by computer for computer communications [JMG81]. This network is now called Cernet. It offers interfaces to most types of machines at Cern and it operates over its own protocols. "It can be considered as a local area network covering the entire CERN site " [JMG86].In fact, from our point of view, the LAN quality of Cernet is not fundamental. Cernet is mainly a backbone for communication and a network supplying application services such as virtual terminal or file transfer.

Cernet is a mesh-structure network offering a datagram service fully guaranteed. It consists of 17 node computers either Modcomp Classic or II/15 computers. The nodes are connected to a high speed data link which uses simple twisted-pair links. The normal data transmission rate is 2.5 megabits/second but over short distances this can be doubled, whilst over long distance (2 km) it may be reduced to 800 kilobits/second to avoid the use of repeaters.

The architecture of Cernet is partially shown on figure 5.4. Three nodes are represented in the circles with their connections to the hosts in the squares. There are now over a hundred of users hosts.

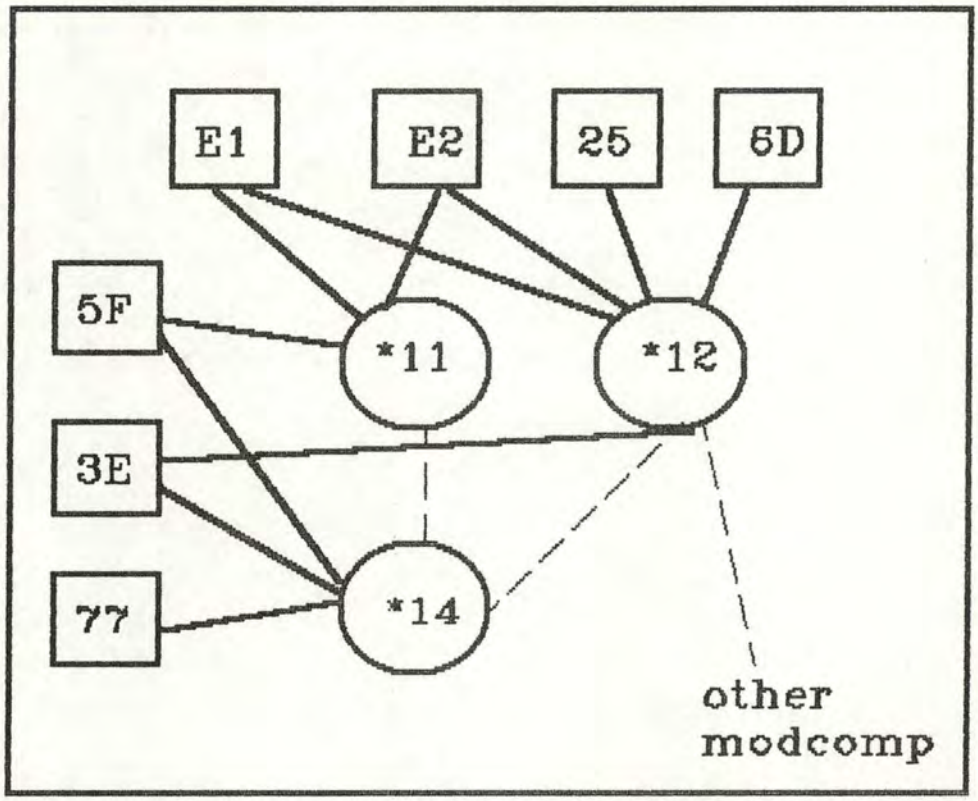


Figure 5.4 CERNET

B. Management requirements for this network

Cernet is at the end of its lifetime and will be replaced within a few years by another backbone (see next point). Giving the dying role of Cernet, it is logical to imagine that only minimal effort will be done to integrate Cernet in the COMS project. This minimal effort will be an interface with the existing monitoring systems of Cernet.

Each node has actually a local monitor task showing the state of the links, the cumulative errors and the packet throughput. All the events are logged in a trace buffer allowing the reconstitution of a crash. A spy program can be run to identify the tasks or regions of code which are spending most time, and to present the results as histograms. The management is done by the control center node.A colour display shows the general network status, particular node status,traffic throughput. So management is done by providing tools to the management staff to see what is happening and act appropriately.

5.1.1.4 The future backbone

Cernet will be replaced by another backbone. This backbone will be at MAC level. The application level which was in Cernet will not be replaced because this can be done by other ways.The study for this network is in progress, therefore only few information are available. This backbone will probably use FDDI technologies over optical fibers. It will operate at a hundred megabits per second over a distance of fifty kms.

Figure 5.5 gives an "OSI like" survey of these networks.
Cernet covers more than three levels, but for this presentation, higher levels are of no interest. There is no need to distinguish IBM and Apollo Domain token ring for the

same reason, the difference between the two networks appearing at higher levels.

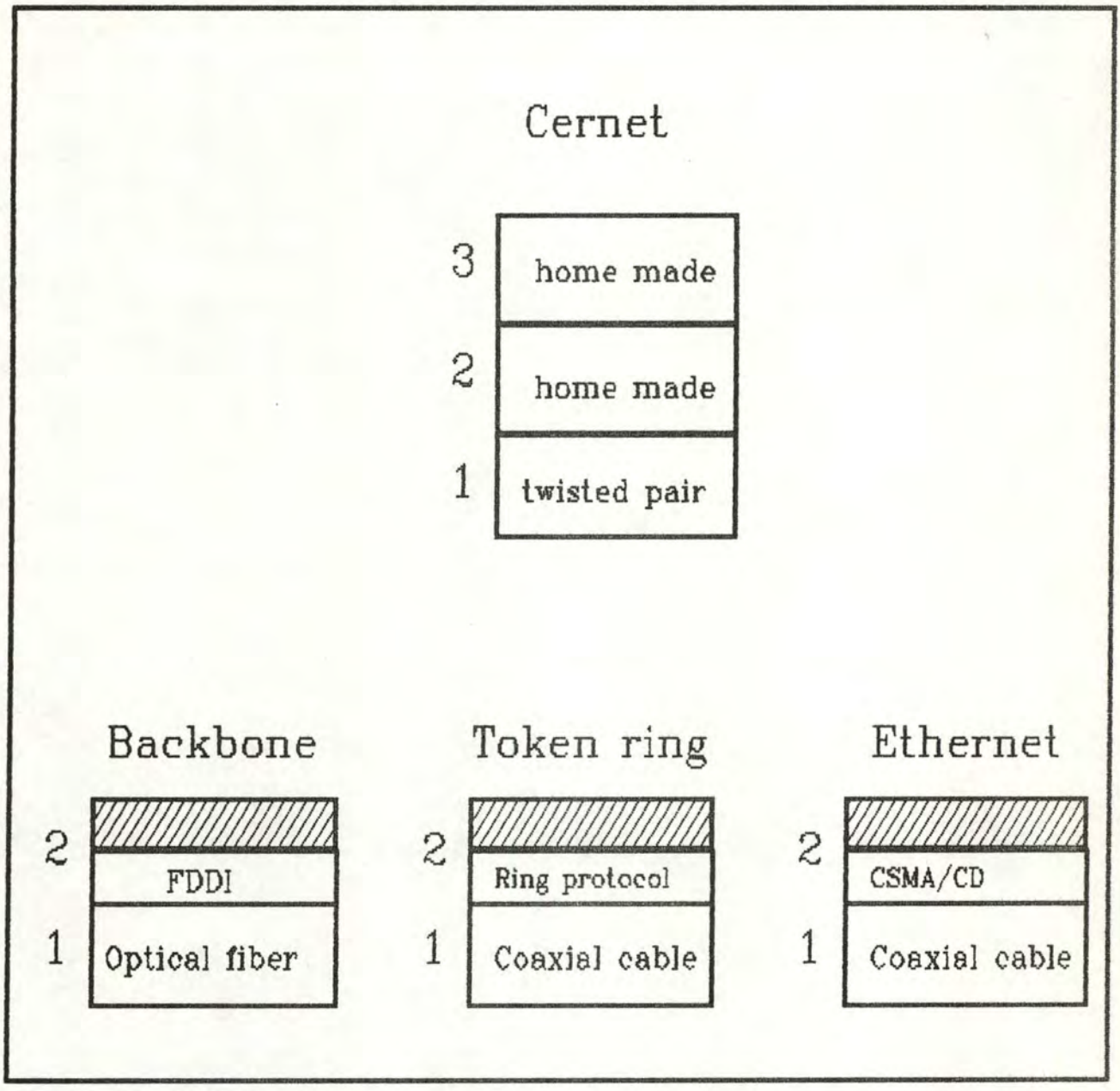


Figure 5.5: First synthesis of the networks.

The reader should keep in mind figure 5.5 to understand the place of the following network: CABAN.

" Mr Andre and Bertrand are connected on the same Ethernet segment, so, they can have a communication, but it is impossible to communicate with Claude or Denis and we need other means of communication".

5.1.1.5 CABAN

A. Architecture of the network

Caban is the Cern Autonomous Bridged Area Network. Its aim is to bridge transparently at a MAC level IEEE 802 series networks (Ethernet, Token ring, token bus).
CABAN uses a set of LAN bridges (Frigate boxes) [PIN86] which are capable of switching inter-network traffic between LANs via a backbone media. The Frigate boxes are designed to permit the incorporation of new LANs and the change to other backbone than CERNET.

Figure 5.6 shows the CABAN architecture: Ethernet segments are connected to frigate boxes, as well as token ring or other LANS.

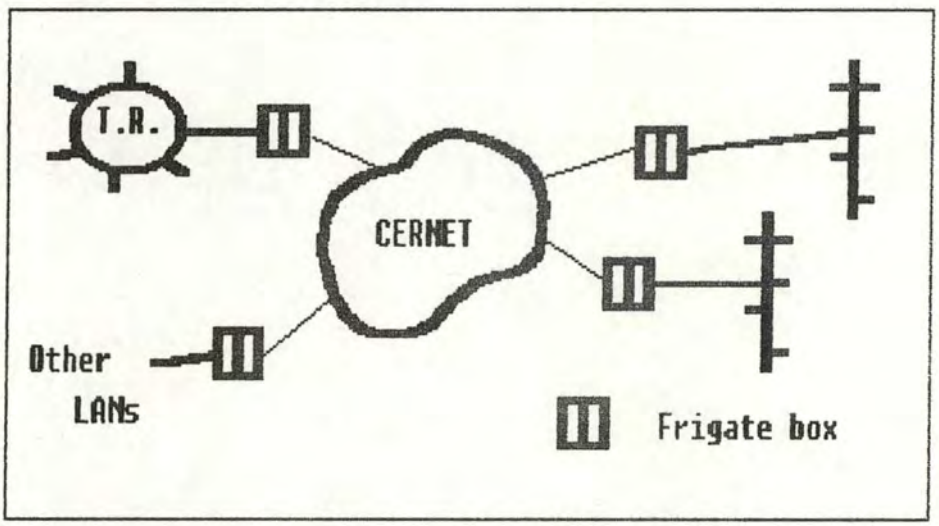


Figure 5.6 : CABAN architecture

B. Management requirements for this network

The COMS project should allow to perform from a remote site all the operations available locally on a frigate box.

Let's do a new synthesis: the service of the Frigate Boxes [DUN86] is a network interconnection at MAC level via a backbone. So in figure 5.7, we have Cernet working

as a backbone at MAC level for the interconnection of Ethernets and token rings (from this point of view, there is no services difference between Cernet and the futur backbone, therefore the future backbone is not represented). The Frigate Boxes are connected together via Cernet by the use of an internetwork layer [Dun86]. The LANs are connected to the Frigate Boxes. So the LANs, the Frigate Boxes and the passage through Cernet form the CABAN network.

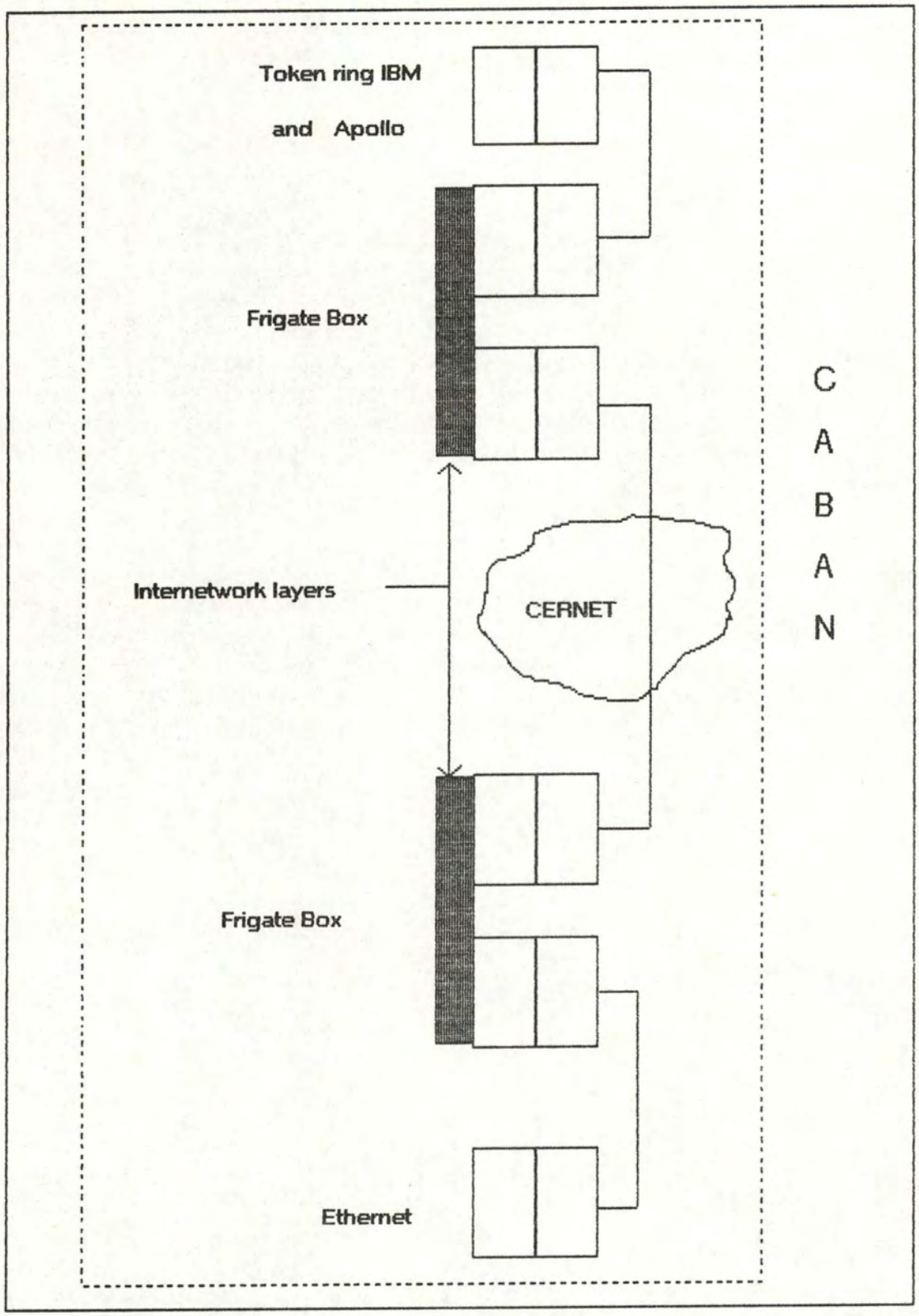


Figure 5.7: Second synthesis of the networks

"Andre, Bertrand, Claude and Denis have a mean of communication between their PCs. But the service provided is very poor and they want to have the capability to exchange messages or files."

CABAN forms a single network including different LANs. This point of view will be adopted by the next network: CERNIP.

5.1.1.6 CERNIP

Cernip is the CERN local TCP/IP service.

A. Architecture of the network

TCP/IP was originally developed for ARPAnet (Advanced Research Project Agency). It covers only ISO level 3 (IP protocol) and ISO level 4 (TCP and UDP protocol). The IP protocol means Internet datagram protocol. It handles the addressing, routing and fragmentation/reassembly of the packets over the multi-media internet. TCP protocol (transmission control protocol) offers a virtual circuit service. UDP protocol (uniform datagram protocol) offers a datagram service. These two services are available to the user.

TCP/IP operates without its own infrastructure.

There are three different types of media in CERN: Ethernet, IBM token ring and Apollo Domain token ring which are interconnected via the CABAN network. There are currently seventy hosts connected [SEG86]. The aim of this net is to "act as a common protocol for the networking of personal computers, workstations and mainframes in the CERN site and to act as an interim transport service (before ISO TP4) for internetworking CERN LANs" [OBR85].

B. Management requirements for this network

The heterogeneity of CERNIP will complicate its introduction in the COMS project. So everything doesn't need to be monitored. The minimum set to monitor is: the essential IP gateways, the main name servers and the main LAN links.

We can realise a synthesis for the logical subset of networks defined at the beginning of the chapter. Figure 5.8 shows the OSI levels one to four. Levels one and two are different for each network integrating them: Ethernet, Token ring, CABAN and CERNET. Levels three and four are the CERNIP network.

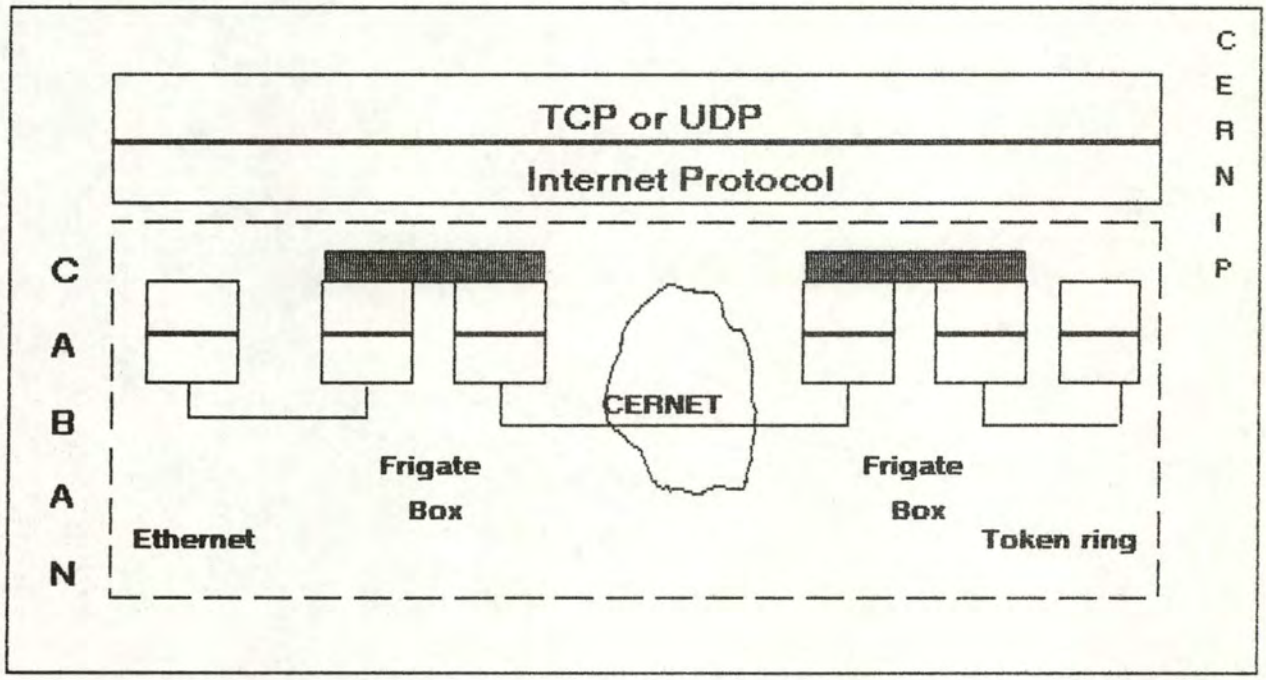


Figure 5.8: Global synthesis of the subset of networks.

"Andre, Bertrand, Claude and Denis can communicate together using the transport service of CERNIP."

5.1.2 EXCITE and DECnet

We will now present a second logical subset of networks composed of EXCITE and DECnet. To make it clear, this subset will be considered as NOT connected with the previous one.
EXCITE and DECnet are two important networks in the CERN in terms of dimension and use.

5.1.2.1 EXCITE

Excite is the CERN local X25 network.

A. Architecture of the network

X25 is the standard developed by CCITT for the communication on a packet switching network.
Level one is based on V24 modems, level two uses the HDLC procedure and level three is an access protocol which provides a virtual circuit service. The X25 program was started in 1980 when it was decided to interface Cernet to the PTT wide area X25 networks. [OBR86] Excite is connected to twenty two hosts in CERN, it offers five connections to public (Telepac, swiss X25) and private (Janet UK, Phynet F, Lep3net USA, Saclay F) networks. Two connections are foreseen to the USA via a high speed satellite connection and to University of Geneva via a high - speed leased line.

The architecture of the network is composed of the following equipments:
- a multigate box which provides all accounting and statistical gathering to the public networks and the user interface for the EXTASE service. (Extase is the private X25 service).
- a CAMTEC switch pad (Camtec 0 in figure 5.9) interfaces the Excite network to the public network and the CERN multigate.The througput is around two hundred packets per second.
- three JNT pads (Camtec 1 to 3 in figure 5.9) form the backbone of the network, providing all the host connections and the majority of the connections to the external private networks. The throughput is around fifty packets per second.
- two JNT mini PAD (Camtec 4 and 5 in figure 5.9) act as X25 concentrators to the main backbone of JNT PADs. The difference with the JNT PADs resides only in the hardware.
- two PADs (CERN PAD 1 and 2 in figure 5.9) to provide an access from the INDEX network (see 5.1.3.2) to the hosts connected on EXCITE.

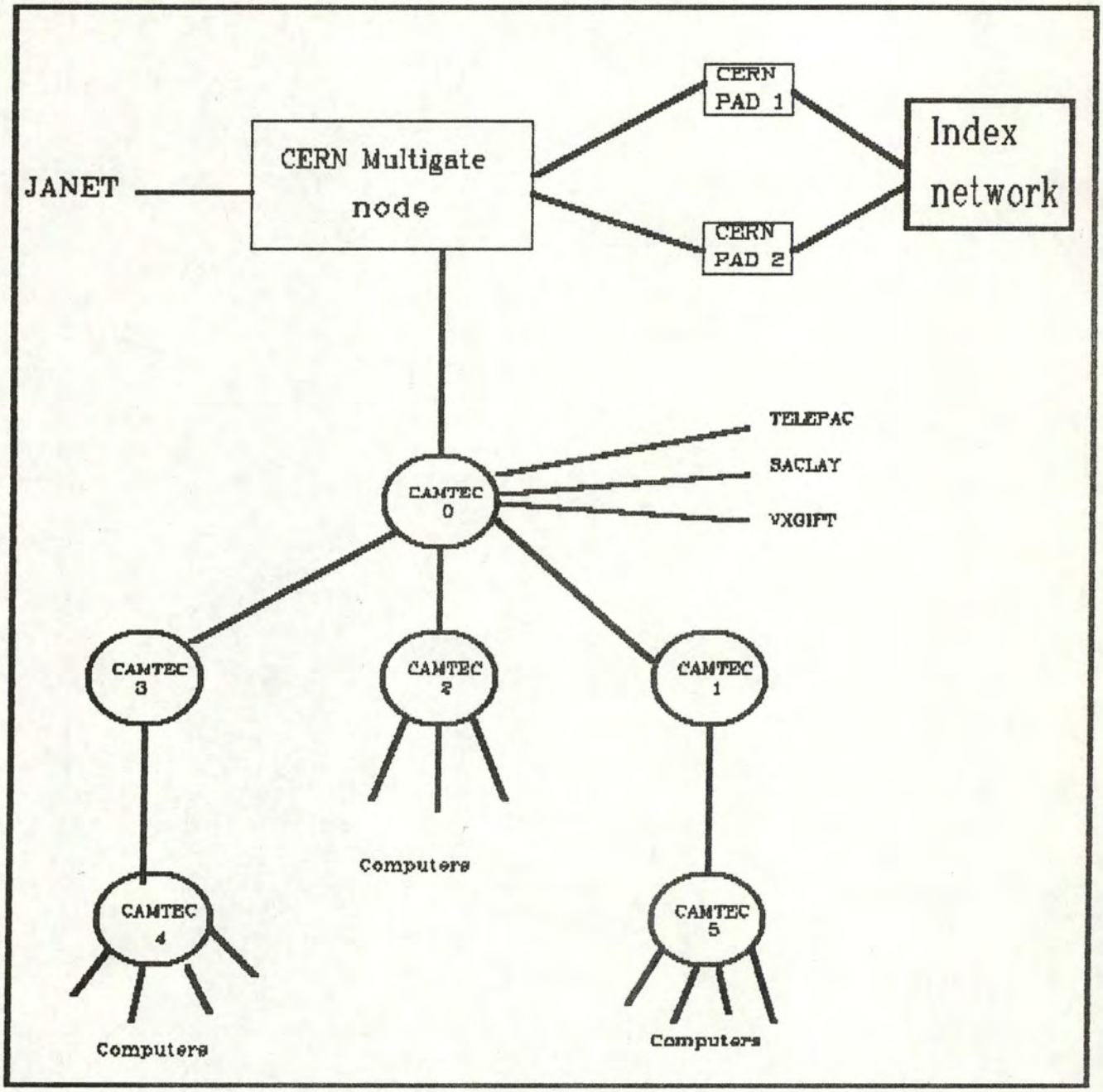


Figure 5.9 Architecture of EXCITE.

B. Management requirements for this network

There are presently some tools avalaible to manage the network. Most hosts connected have facilities for logging line traffic for off-line analysis. There are

two media analysers : the first one is simple line monitor and the other one allows in depth on-line and off-line analysis. There is a network monitor to poll the nodes of the network. It is planned to introduce a new switch to replace the existing set of small switches. This will totally replace all the functions of the multigate [OBR86]. This new switch will provide a network management protocol to which a management system may be attached. This will be integrated into the COMS project.

5.1.2.2 DECnet

A. Architecture of the network

DECnet is the proprietary architecture developed by Digital and originally implemented on PDP 11s, DEC 10s and 20s and later on VAX/VMS machines.[HEI86] It provides a comprehensive set of services for DEC machines.

The name DECnet is also given to the host switching network which can work over different media: Ethernet, DDCMP, X25, point-to-point links. It is mainly a LAN but it has been enhanced for wide area operations. In contrast to other proprietary architectures, DECnet has a certain degree of openness. It works over non-DEC machines or operating systems. There are now about one thousand hosts connected to DECnet and fifty in CERN. So it is both a LAN and a WAN. The interconnection to other DECnet areas is done via a level two router (L2R). (See figure 5.10)

In CERN, DECnet is mainly based on Ethernet, but a few DDCMP links (Digital

data communications message protocol) and X25 links exists.
DECnet is now migrating for the use of OSI protocols.

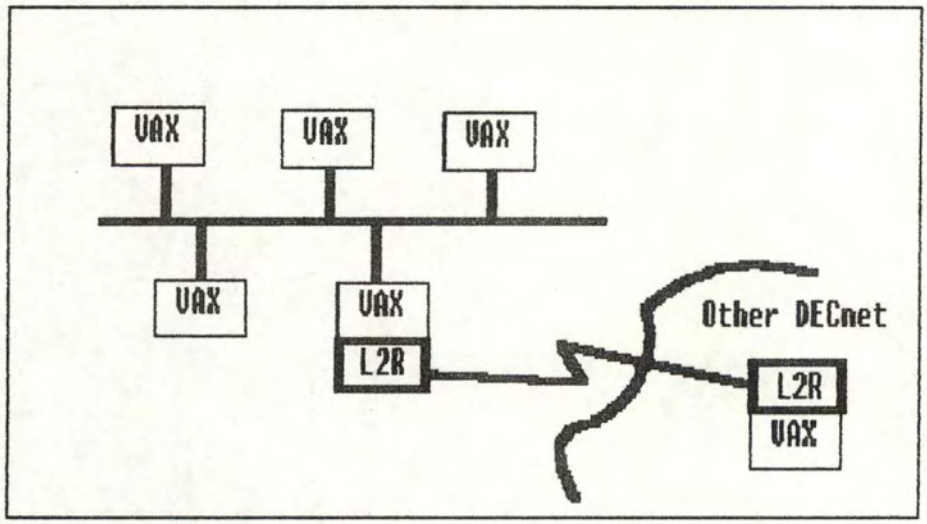


Figure 5.10: DECnet topology.

B. Management requirements for this network

Some management tools are provided by Digital Equipment to monitor DECnet. They should be integrated in the COMS project.

We can do a synthesis to show the way DECnet works on EXCITE, Ethernet and DDCMP. Figure 5.11 shows the different OSI levels of DECnet. Levels one and two can be X25 (V24 and HDLC), Ethernet (Media connector and CSMA/CD) or DDCMP (DEQNA connector and DDCMP). Levels three to seven are provided by DECnet.

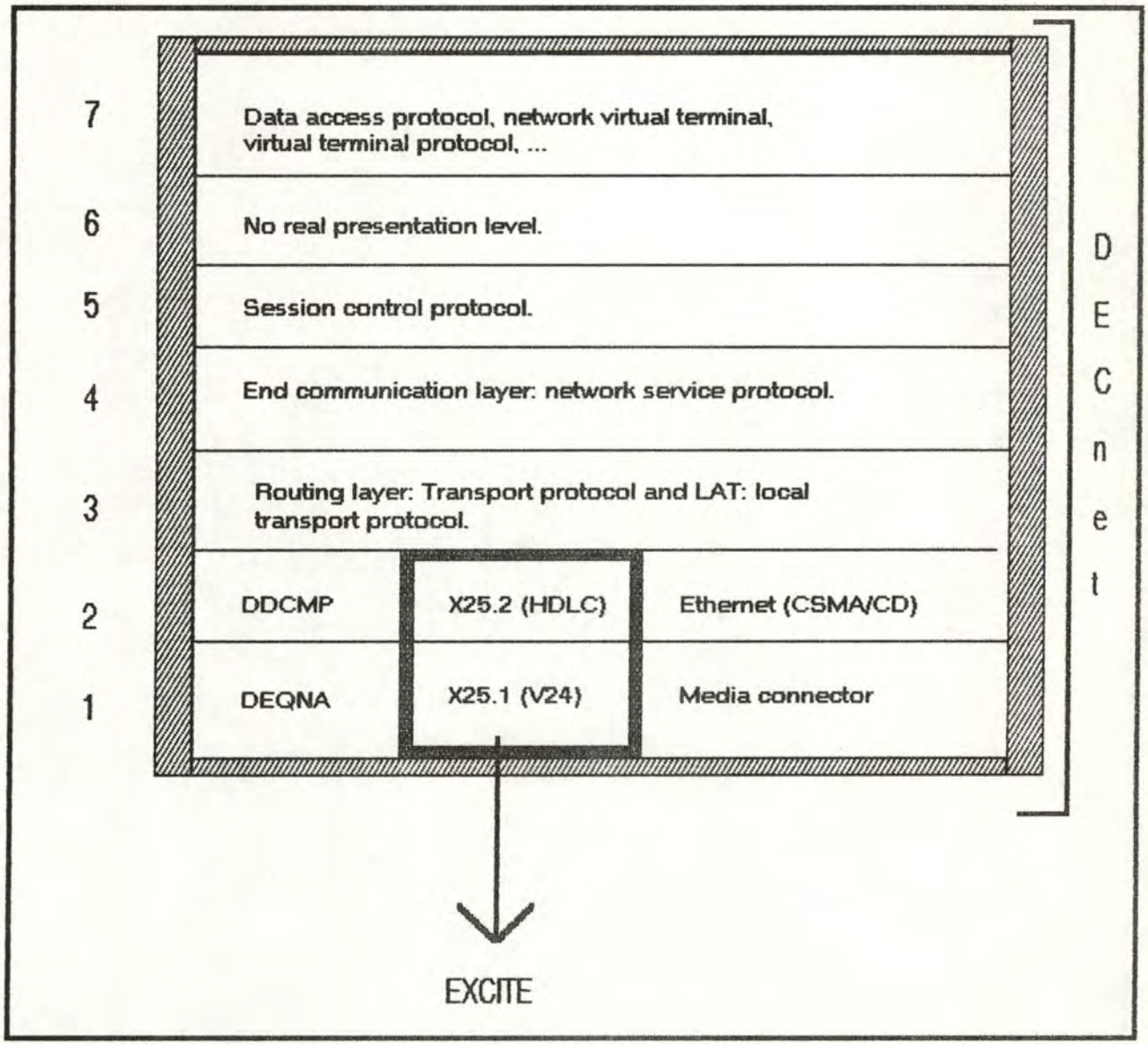


Figure 5.11: OSI-like survey of Excite and DECnet.

5.1.3 Other networks of CERN

We can now present SNA, INDEX, EARN and UUCP. They will not be developed in detail because they are not fundamental to the comprehension of the high level of heterogeneity and of interconnection in the CERN networking environment. The interconnection of these networks with the networks of the two logical subsets is considered as non existent.

5.1.3.1 SNA

SNA is the system network architecture developed by IBM in '75. It is the IBM's standard for both local and wide-area networks. It defines the interconnection of all IBM type devices from terminals to hosts. Primarily, it operated over SDLC physical links although later enhancements have enabled it to run over other media. The open links are only permanent circuits.

In the CERN context [FOS86], SNA is used to provide communication between the IBM mainframes and other proprietary LANs, and also communication with external institutes. But as it is only based on permanent circuits, SNA is costly over public wide-area networks such as X25 or ISDN.

The topology of the network can be seen as a set of trees (see figure 5.12) which forms host domains. The front-end processor (3725 in figure 5.12) has a monitor console which is directly connected to the equipments.

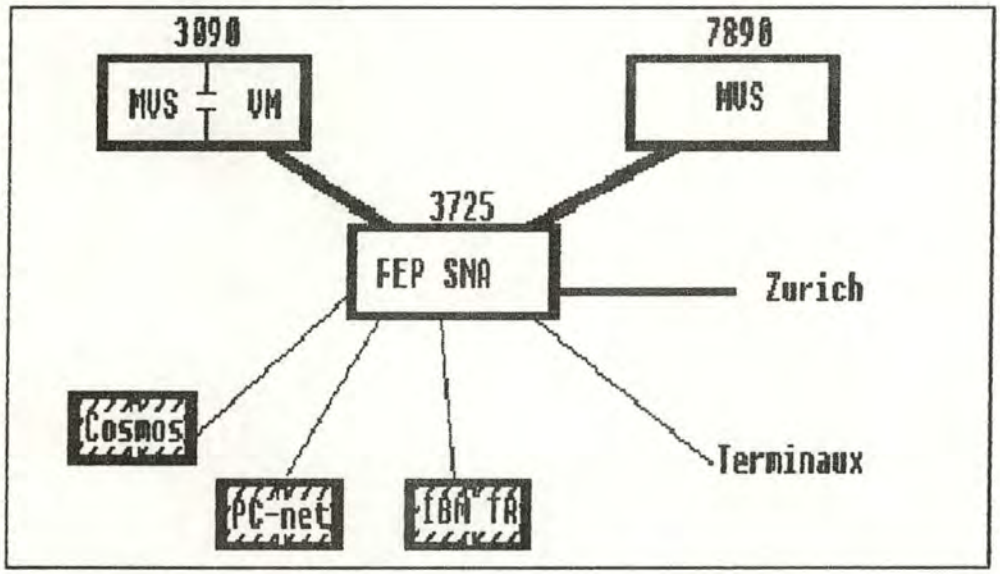


Figure 5.12 SNA architecture.

Figure 5.12 shows an example of SNA architecture. A 3090 IBM mainframe running two operating systems: VM and MVS and a 7890 IBM mainframe running MVS are connected to a front-end processor 3725 . This 3725 is connected to three LANs: Cosmos for the Norsk Data computers, PCnet for the PCs and IBM token ring. It is also connected to Zurich via a modem and finally the terminals are connected.

5.1.3.2 Index

Index is a CERN wide public use data transmission circuit switched network. It was primarily used for the interconnection of dumb terminals to a selected host. All the circuits are of the type RS232 C and the selection is made through an interactive dialogue.

A. Architecture of the network

The network is based on Gandalf switching equipments (PACXes) distributed in the CERN. There are different types of Gandalf equipments: PACX switching nodes, PACX oriented data sets and Gandalf limited distance data sets. [BRO86]
There are different types of modems: Data over voice modems, dial-in modems, X25 modems.
The service now permits the connection of dumb terminals, the connection of PCs for file transfer, program downline loading... , and the connection to external networks such as Rutherford Lab., IN2P3 Paris, LAPP Annecy or X25.

B. Requirements for this network

The service is 24 hours/day, all year round terminal switching service. There are at the moment no network monitor or management tools. So it has been preferable to develop these tools to maintain the service. They will be developped in concordance with the COMS project.

5.1.3.3 EARN

The European Academic Research Network is a free international network supported by IBM. It is a wide area network using leased PTT lines at 9.6 kilobits per second. It is connected to his US and Canadian homologues respectively BITnet and Netnorth. The three networks appear as a single logical addressing space for the users. At the moment, EARN is using Binary Synchronous Protocol (BSC) at the link level [MAR86]. This network provides a wide set of services: Electronic mail, file transfer, remote job entry, file and database servers, distribution list servers, conferencing... but no remote terminal access. EARN is connected to the MINT gateway (see section 5.2.1) to allow the sending of mails to UUCP and otherwise.

5.1.3.4 UUCP

UUCP is a network developed to connect hosts running UNIX systems. It provides file transfer, electronic mail and remote command execution.
UUCP is mentioned in this description because the MINT gateway (see 5.2.1) connects the EARN-BITnet environment with the UUCP environment.

5.2 The CERN gatewaying environment

The role of gateways is primordial in the networking environment of CERN due to the number and the heterogeneity of people coming to experiment on the accelerators. So it is useful to give a hierarchy of the different gateways in CERN. Bob O'Brien [OBR85] explains: "The term "gateway" can be divided into four distinct types: application level gateways, transport level gateways, network relays and bridges". We will use this distinction to present the different "gateways" available in CERN.

5.2.1 Application level gateways

The aim of an application level gateway is to interconnect specific applications such as Electronic mail, file transfer or network management. The user can so use services on different networks. In CERN, the application level gateways are GIFT, MINT and FAID. The COMS project is an application level gateway too.

A.The GIFT gateway

General Internetwork File Transfer is a gateway which allows the high level translation and gatewaying of file transfer protocols.
GIFT consists of a GIFT kernel which implements a GIFT common meta-protocol and of several protocol slots (one per type of protocol) which interface a given file transfer protocol with the kernel [FLU85]. Figure 5.13 shows the architecture of such a system.

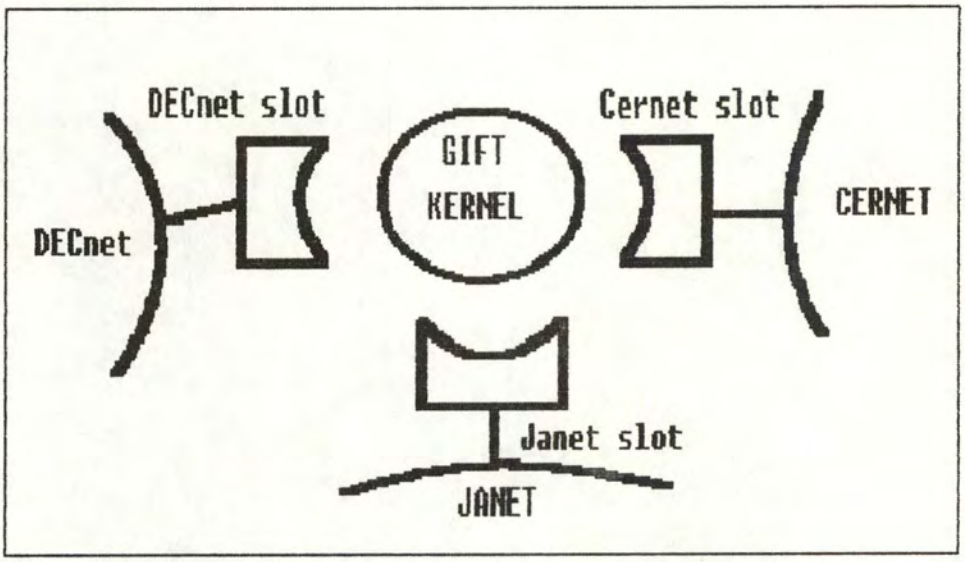


Figure 5.13: GIFT

B: The MINT gateway

Mail Interchange (MINT) is a high level gateway for Electronic mail. It changes the format of the mails to interchange them between different networks.
Figure 5.14 shows the architecture of such a system, it is similar to GIFT but the internal protocols are different.

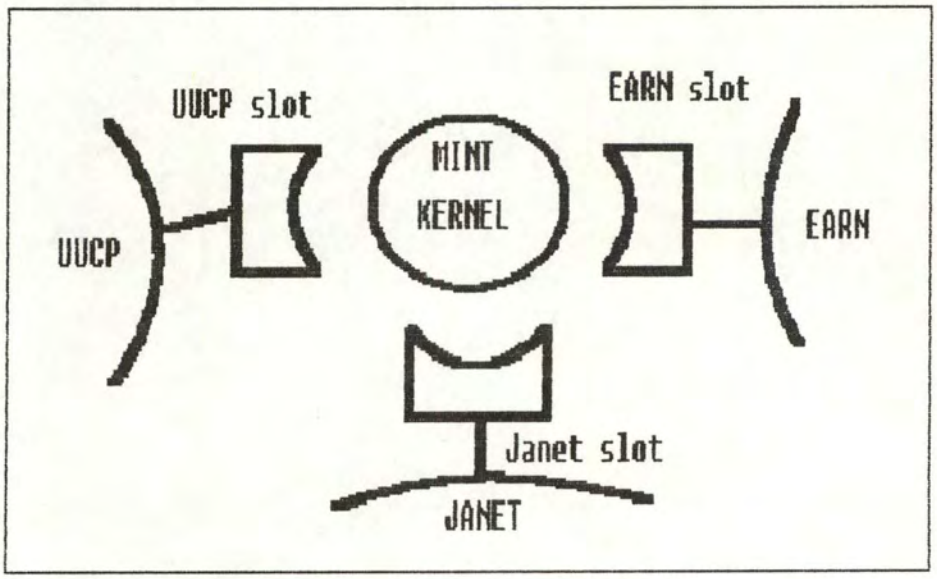


Figure 5.14: MINT.

C: The FAID gateway

The File AID service operates over the CABAN network. It is a file transfer gateway between the CERNET protocol and the simple "FACS" protocol defined on Ethernet.

5.2.2 Transport level gateways

The aim of the transport level gateways is to provide the transportation of data independently from the network service.

Example: - DECnet over X25, Ethernet and DDCMP.
- ISO transport level gateway.
- Yellow book transport service over X25 and Ethernet.

5.2.3 Network relays

The network relays interconnect one network to another with the goal of providing the intersection subset of the two network services to each other and of using the network service of one in the infrastructure of the other.

Example: The IP gateway between the token ring and the Ethernet.

5.2.4 Bridges

Bridges are gateways at the lowest level. They give the interconnection of one network media to another. They operate at the Medium Access Control (MAC) level. The CABAN network uses bridges.

5.3 The concept of network management in the CERN

We will present in this section the results of the work of Bob O'Brien in the CERN about network managament. We will compare these concepts with the recommendations of OSI in section 5.6.

"Network management is the management of network resources to maintain the grade of services as specified in the network service definition for this network" says Bob O'Brien [OBR85]. It is evident that this definition can be broken down in different types of management following a time line. The long-term decisions will be called the network policy management, the every day maintenance of the services will be called the operational management and the 24h observation of the network is the network monitoring.

It can be interesting to examine the structure of the people who will take these decisions.

Figure 5.15 shows the hierarchy of this structure. The chairman is the ultimate responsible for all decisions, the service managers are responsible for an individual service,the operational coordinator is the mediator between the communication support committee and the operators,the user consultancy office (UCO) representative has the same role for the final users.

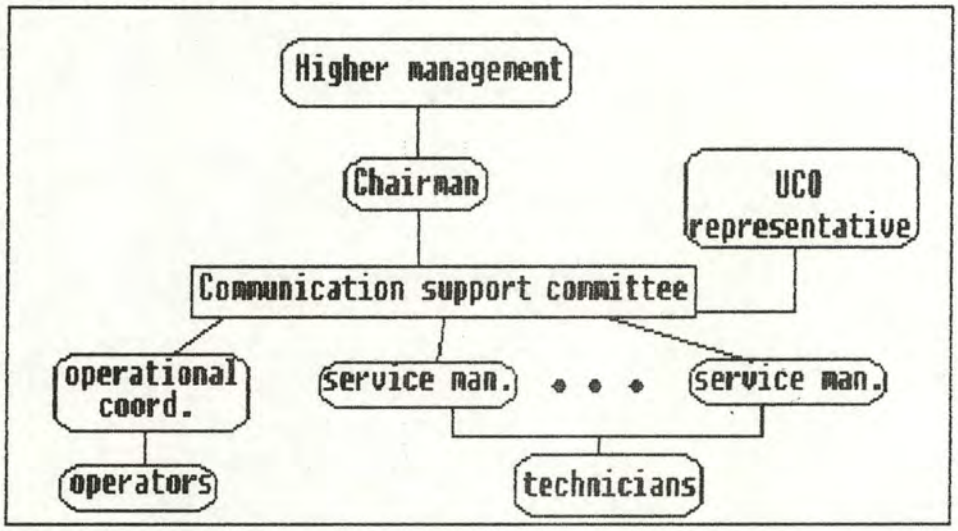


Figure 5.15: Management structure.

5.3.1 Network policy management

This part of the management deals with the control of the network as a whole. It covers all major decisions for a given network about the following points:

1: All decisions affecting major alterations to the network: this will include the major network reconfigurations such as adding new subnet nodes, coordination of protocol changes such as migration to ISO for CERNIP and implementation of new routing philosophy.

2: All decisions in progress to take actions on major alarms/faults which are not in the domain of operational management. This includes for example the reaction to a network overloading, to a general long term service degradation.

3: All decisions affecting the setting of parameters and constraints for the operational management of the network for example: the setting of the criteria for new node installation on the network and address allocation.

4: All decisions affecting the planning for the evolution of the future network: e.g. the evolution of Cernet to the futur backbone.

5: All decisions affecting the definition of interface mechanims to the operational management.

All these decisions are taken by the overall coordination body which includes all the people in charge of management (see figure 5.12).

5.3.2 Operational management

The operational management will receive the parameters decided by the network policy management and will be responsible for the everyday maintenance of the service in the network. The following actions are within his competence:

1: Maintenance of the network monitoring subsystem: this means, for example, making sure that the topology schematic is up-to-date.

2: The allocation of resources within the parameters given by the network policy management: connections of new users or of news hosts.

3: The fault handling and the documentation of non severe network faults: bad physical line investigation, informing operators of change, ...

4: Minor network reconfigurations: switching in backup nodes,...

5: The communication to the network policy management of all major problems which can affect the long-term grade of service: e.g. the risk in medium-term of overloading in the network.

6: Initiating of fault recovery and network maintenance.

These actions are taken by the service managers helped by the technicians.

5.3.3 Network monitoring

The network monitoring is a passive process. It is typically the role of a computerised system to assume the following functions:

1: To accurately present the topology and the state of the network as a whole, focusing on object which is critical for the maintenance of the grade of service.

2: To present the meaningfull alarms to the operational management staff.

3: To filter out minor alarms until they have occurred in sufficient quantities to require actions.

4: To maintain statistics of network operations and operational reliability.

The hierarchical structure of this model permits each layer to assume its job without being troubled with the problems of the lowest levels.

5.4 Network management within a multi-network environment

It is evident that when more than one network is present, the above description is not sufficient. The situation is more complex because the management will need an overall coordination, as well as to monitor the inter-network traffic via a monitoring of the gateways.

5.4.1 Overall coordination

When introducing an interconnection for the networks, it becomes necessary to add a new layer to the previous model. This layer is called the communication coordination and is made by the communication support committee (see Figure 5.15 at the beginning of section 5.3).
Its functions are the following [OBR85]:

1: To control the development and the installation of new networks.

2: To coordinate the requirements and the feasibility study for any new communications project.

3: To influence policy decisions made by the network policy managers to avoid the duplication of effort.

5.4.2 Monitoring of gateways

The gateways and some communication items provide services to more than one network. They provide services above that of any individual network, they must be treated so as a special case. We can apply the preceeding model to these objects:

1: Policy management: The policy managers are involved when they are affected by any changes in the service offered by the gateway.

2: Operational management: It will be necessary to define operational management procedures to monitor each type of gateway.

3: Monitoring: The gateways are connected to more than one network, so they will be apparent on the monitoring system of each network they are connected to. They will require special monitoring functions depending their functionnalities: e.g. Is MINT interchanging mails correctly? Are the IP gateways internetting IP packets correctly?

5.5 Network management system

It is clear that network management systems will consist of both manual and automated procedures. A network management system is [OBR85] " an automated entity which aids in the management of communication systems." This system is placed over the monitoring of the network and takes some functions of the operational management. This system will be used mainly by the operators and by the service managers but some functions will be available for the users.

5.5.1 Functions of a network management system

It is clear that a network management system must help the manager by giving him a clear presentation of the management information and a common interface to interact with all the sytems. The functions of a network management system can be classified

into three categories [OBR85]: the management facilities, the data management and the communication system interface.

A: the management facilities

The network management system must include some facilities to help the user in his management function. Here are some examples of facilities which can be include:

1. There is a multitude of alarms appearing on a network. All of these are not crucial for the manager, it is interesting to integrate a tool to filter the high level alarm which affect the operation of the communication system.

2. From the same point of view, a tool can give a dynamic classification of the alarms into different categories e.g. :

- emergency alarms: which can affect the operation of the communication system.
- urgent alarms: which can create an emergency alarm.
- warnings: which can be interesting for a manager when nothing urgent occurs.
- low level alarm: which can be interesting to observe during a period of time.

3. During the test, the development or the maintenance, the system must provide the ability to isolate an area.

4. The manager must permanently control the systems under his responsability. The activity must go on independently of other activities.

5. The management of communication resources is the first function of a management system. So it must be possible for the user to book or allocate communication ressources via the management system.

6. The system should help the manager to determine and isolate the problems. This must be done via the internal knowledge of the system.

7. It must help in the tracking of the problems from their onset to their solution. So future problems should be solved using these references.

8. It must produce reports for the different types of people involved in the management. These reports will different for the service manager or for the operator.

It is possible to imagine other facilities which can be introduced into the management system.

B. Data management

The management system will have to access various information. These data will reside in its own database or be retrievable from a remote source. It is necessary to keep these data up-to-date. For these reasons, it is useful to introduce data management facilities in the management system. These facilities can be:

1. A local database management system for all the information kept locally by the management system.

2. A data manager to accept data from a remote source.

3. A data manager to enable the collection of information of external data when required.

These facilities will give the ability to manage a distributed database.

C: Communications systems interfaces

It is evident that the management system will have to interact will a wide variety of communication objects. It has, therefore, to include a highly flexible range of interfaces. The interfacing does not only happen at physical level, but also with network monitor or network management tools. It is possible to define different types of interactions[OBR85].

1. Raw interaction: If the network to be monitored does not offer any monitoring systems, we will be in a raw interaction. It means that the network management system will have to provide all the management facilities for the network.

2. Interaction with network monitor: If the network already has a network monitor developed following its own requirements, it may be cost-effective to interconnect this

monitor with the network management system depending on the existence of appropriate piers to extract information.

3. Interaction with a management system:If a system already has a management system, it can be interesting to exchange information at the higher level between the two systems.

5.5.2 Architecture of a network management system

The architecture which will be presented, has been developed in the COMS project to permit the sizing of the databases and to make some assumptions about the implementation of the end system. Figure 5.16 gives of graphical representation of the system.

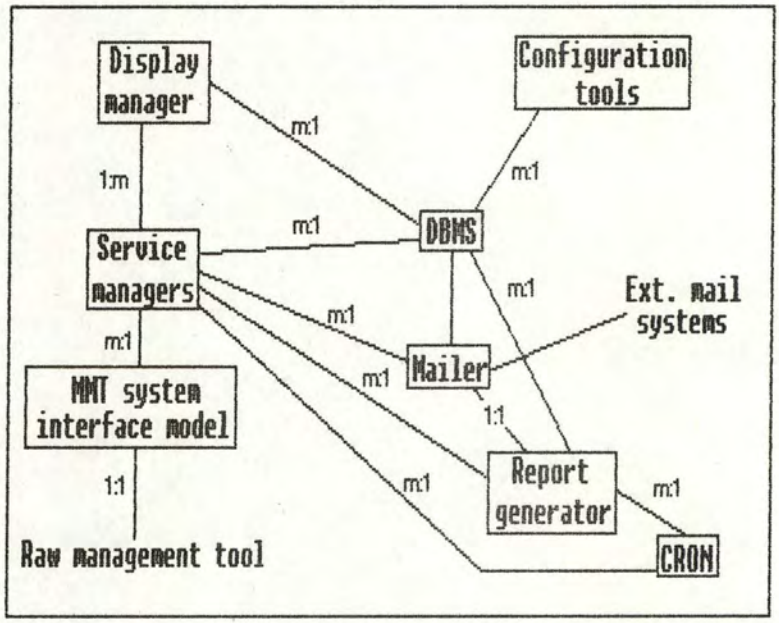


Figure 5.16: Implementation model.

Rectangles represent processes and lines represent the processes relationships. The numbers on the relationships are the connectivity. All these processes will now be defined.

A: The display manager

The function of the display manager is to present the management information to a user. The characteristics of the display manager depend on the device type used (e.g. a terminal or a graphics workstation), the type of interface required by the user and the type of user requesting the display manager (e.g. an operator or a normal user).

B: The service manager

The service manager is responsible for all aspects of service management. It handles all the interactions required by a specific service. It will interact with the DBMS to access all the necessary information, with the mailer and the report generator to obtain specific services and with the Cron to obtain timing information.

C: The management system interface module

The management system interface module is responsible of interfacing the raw management offered by a system with the internal framework. Due to the diversity of interfacing requirements, each interface module will be sub-system dependent.

D: The database management system (DBMS)

The DBMS is responsible for the management of all local or remote information.

E: Configuration tools

It is necessary to have internal tools to configure the COMS project itself. All these tools (service definition tool, display configuration tool) are situated in the configuration tool.

F: Mailer

The mailer will enable the following functions: submission of management request (report generation,...), delivery of semi-urgent reports, delivery of regular reports, convocation of a group of personnel (e.g. the persons in charge of the Frigate project).

G. Report generator

Due to the diversity of users, there will be different types of reports. Wherefrom the necessity to use of a report generator which enables to easily specify the format and the content of reports.

H. Cron

Some activities must be scheduled (report generation,...): so the COMS project must have some tools to plan activities; they will be located in the Cron module.

5.6 Concordance of the COMS project with the ISO model

The main points of concordance are the following:

- The COMS project is a management presentation service. It can use the management facilities offered by the SMAP process within any open system managed by COMS.
- The management functions defined in the ISO model are a superset of the management functions provided by COMS.
- COMS will provide the user services specified in the ISO model.
- Specific open systems and their SMAPs will be individual subsystems in the COMS service model.
- The protocols defined by ISO for the exchange of information will be used by the COMS project as soon as they will be defined.
- The principle of management information base is included in the COMS project. But it will be distributed between different machines.

The main discordance are the following:

-The COMS project will support other subsystems than the subsystems depending on the open system domain.
- ISO says nothing about the presentation to the users, COMS will devote important effort in this area.
- The definition of services for COMS covers the support of grouping of subsystems into operational services. This is wider than the ISO point of view which covers only the support of

physically defined open systems.

- The COMS project will offer services to manage application level gateways and application level services.
- The definition of services for ISO is based on the class of services: configuration management,... on the COMS project, the services are sorted in terms of time: operational management,...
- The layer management used by COMS will not be as complete as in the ISO model. The reason is that the CERN networks are not developed under the seven layer model.

5.7 Our participation to the project

The COMS project officially began on the first of January 1987. As we arrived in CERN on the first of August 1986, we were involved in the specification and the feasibility studies.

Our main job in the project was a feasibility study concerning the display manager. The goal of this study was to test two products in real conditions. It means that we had to produce test programs to analyze the facilities provided by the two products

The first product is a communication tool working with TCP/IP. It was planned to use this product for the communication between the different modules of the architecture defined in the precedent section. This product is called IPC (Inter Processes Communication) and is developed by Apollo. It is based on the notion of socket developed in UNIX Berkeley version 4.2. A socket is [LEF83] " an endpoint of communication on which some primitives can be used. Each socket has a type and one ore more associated processes within a communication domain." The different domains are the UNIX domain and the INTERNET domain. The domain influences the format of the sockets name and address. The different types are stream socket, datagram socket and raw socket. The primitives associated to a socket are: socket creation, binding a name to a socket, make connections between sockets, transfer data on these connections, closing connections and I/O multiplexing.

The Apollo workstations run a operating system called AEGIS. This operating system is, in fact, the concatenation of an old Apollo operating system with UNIX BSD 4.2. Therefore the good running of IPC was not evident and we decided to test first the primitives for socket manipulation on VAX 780 running UNIX Berkeley 4.2. The aim of this first test was to communicate between two processes located on the same VAX. One process was the sending process and the other one was the receiving process. After this test, we transferred the programs to the Apollo without changing anything to

see the conformity of AEGIS to UNIX BSD 4.2. The results were very good in terms of speed, error rate and communication establishment except for UNIX domain that seems very badly implemented on the Apollo. We never managed to establish a communication for the UNIX domain. The next step was to communicate between two processes located on different machines. For that, we transferred a sending and a receiving process on the VAX and the Apollo, and the VAX-sending process communicate with the Apollo-receiving process and vice versa. Once again the results were very good in terms of speed, error rate and communication setting up. Finally, we placed a receiving process on the Apollo to communicate with sending processes located on the VAX and on different Apollo workstations.
The detailed results of this study are provided in the Annex A in the report called: "Report over the use of IPC on the Apollo".

The second product tested was a tool for defining the interface with an application program and specifying how the interface should be presented to the user. This tool is called DIALOGUE and is developed by Apollo. This tool can be divided into two different interface definitions: the user interface definition and the application interface definition.
- In the user interface definition, the programmer will define the way the program will interact with the program. The programmer will allow some input actions (e.g. to click with the mouse in a special window) and will structure the ouputs using all the power of the Apollo display manager (e.g. the use of windows, ...).
- In the application interface definition, the programmer will define the way the program must react on the tasks triggered off by the user interface.
The link between the user interface and the application interface is shown on Figure 5.17. The user interacts with the user interface to notify to Dialogue he wants to trigger off some tasks, Dialogue interacts with the application using the application interface, the results produced by the application are provided to Dialogue by the application

interface and Dialogue provides these results to the user via the user interface.

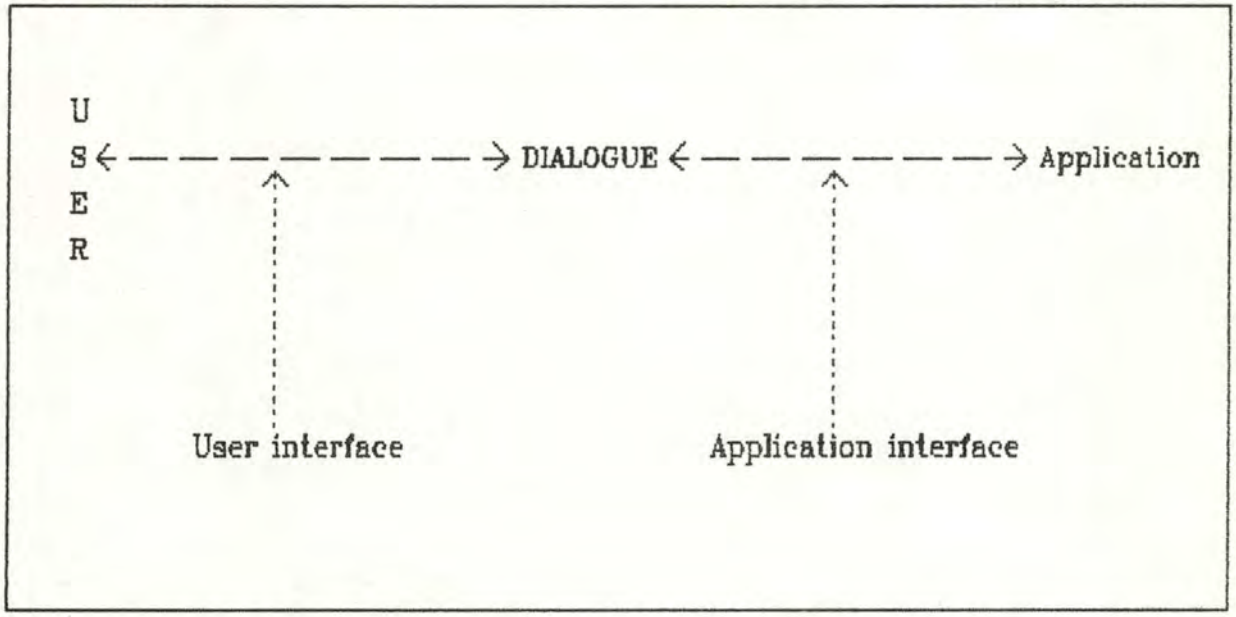


Figure 5.17: Relationship between user and application.

With this tool, we made a program to present graphically the architecture of a network.

The report called " Report over the use DIALOGUE on the Apollo" in annex B includes the presentation of the product, the presentation of some typical algorithms and some tricks to use the product.

The COMS project is a very ambitious project. It was only at its beginning when we left CERN, therefore, it is difficult to give a critical judgment on it. But the feasibility studies were positive. We made a small simulation program using the tools presented above and we did not find any incompatibility. The main problems encountered by the project are until now "political problems" between Digital and Apollo. At the beginning of the project, it was planned to have Digital and Apollo cooperating with CERN to develop the project. But after long negotiations, Digital has decided to stay alone with CERN.
This participation of Digital is a stimulating sign for the future of the project by giving to it a commercial aspect that is not existent inside the CERN.
The COMS project is also interesting for other reasons then network management. It has forced the responsible of each network to give a precise definition of the services

provided and to search the complete architecture of the network. It has also created links between network managers that will be used for other projects.

So, the COMS project will have many effects on the networking organization of CERN: these effects will be on the infrastructure, on the knowledge and on the managers of the different networks.

6. Conclusions

The network management is a problem that becomes more and more important for all people developing or using networks. The complexity, the heterogeniety, the interconnectivity of the networks is fastly increasing. This situation leads the network managers to adopt a behaviour which can be illustrated with the sentence: "it works, do not ask me why!!!". There is an urgent need for network management systems. ISO has begun to answer this need by giving the first drafts about the OSI network management. But until now, there does not seem to exist management system for a widely heterogeneous and interconnected networking environment.

The CERN is trying to fill up this gap by developing the COMS project jointly with Digital Equipment Corporation.

We have presented the present level of standardization for ISO concerning network management. This approach is academical and very wide but it is limited to the open systems.

This approach is not sufficient for a networking environment with various different networks interconnected. All systems in such an environment are not open systems, but they need to be managed too.

In the COMS project, the approach is wider than for ISO by allowing the management of all the main CERN networks.

A lot of questions remain unanswered about the OSI management or the COMS project. The future development will give the response to them but we can try to give some indications about the main question which is: "Will the COMS project be an operational multi-network management system?" All the indications until now are favourable:

- The feasibility study seems to give positive results for the programmation tests and the sizing of the database.

- Digitial Equipment is a full partner in the project and they have detached three engineers to work with the CERN people. Digital does not have the reputation to waste money and time for a project with no future.

- The environment provided by CERN is very good. It means that financial constraints are not as important as in private institutions.

We think that in the future, we will see an important development of projects similar to the COMS project due to the increase of the strategic aspect of networking for a lot of people (bank, army,...). The developing need of communication between users of any community of interest will increase too the necessity of management systems due to the increasing of the connectivity between international networks. The freezing of the change in the networking environment due to the refusal of information technologists to modify their habits is the last reason, this freezing implicates the simultaneous existence of old and new networks.

References:

[BRO86] Brodmann, H., Cannon, S., O'Brien, R.P., Sohet, L. (1986). 'Network information request for INDEX'. CERN/DD/COMS-REQ/G10.

[DUN86] Dunon,P., (1986),'A contribution to the network interconnection problem'. Work of 1986.

[FLU85] Fluckiger, F., Valente, E. (1985). 'GIFT project: Status report mid 1985'. CERN/DD/EDC-FIL/S5.

[FOS86] Foster, D., O'Brien, R.P. (1986),'Network information request for SNA'. CERN/DD/COMS-REQ/G7.

[GRE83] Green, P.E. (1983),'Computer network: architectures and protocols'. Plenum press, New-York.

[HEI86] Heiman, G. (1986),'Network information request for DECnet'. CERN/DD/COMS-REQ/G14.

[I0382] ISO,(1985),'Procedures for management information service standardization'. ISO/TC97/SC21-N382.

[I0391] ISO,(1985),'Sixth working draft on basic management framework'. ISO/TC97/SC21-N391.

[I0980] ISO,(1985),'Security management service definition'. ISO/TC97/SC21-N980.

[I0981] ISO,(1985),'Accounting management service definition'. ISO/TC97/SC21-N981.

[I0982] ISO,(1985),'Configuration management service definition'. ISO/TC97/SC21-N982.

[I0983] ISO,(1986),'Performance management service definition', ISO/TC97/SC21-N983.

[I0984] ISO,(1986),'Fault management service definition', ISO/TC97/SC21-N984.

[I1373] ISO,(1987),'Common management information service definition', ISO/TC97/SC21-N1373.

[I7498] ISO,(1984),'Open system interconnection, basic reference model', ISO/TC97-IS7498.

[I8648] ISO(1986),'Internal organization of the network layer', ISO/TC97-DIS8648.

[JMG81] Gerard, J.M. (1981),'CERNET-A High Speed Packet-Switching Network', CERN 81-12.

[JMG86] Gerard, J.M. (1986),'Network information request for Cernet', CERN/DD/COMS-REQ/G13.

[LEF83] Leffler, S.J., Fabry, R.S., Joy, W.N. (1983),'A 4.2bsd Interprocess Communication Primer', Draft at july 27 1983.

[MAR86] Martin, O. (1986),'Network information request for EARN', CERN/DD/COMS-REQ/G11.

[OBR85] O'Brien, R.P. (1985), 'Network management at CERN: A basis for discussion', CERN/DD/EDC-GEN/V6.

[OBR86] O'Brien, R.P. (1986),'Network information request for the CERN X25 network EXCITE', CERN/DD/COMS-REQ/G6.

[PET86] Petrilli, A. (1986), 'Network information request, Apollo Domain', CERN/DD/COMS-REQ/G8.

[PIA80] Piatkowski, P. (1980), 'The ISO-ANSI Open systems reference model: A Proposal for a systems approach', Computer Networks 4 (1980), pp 111-124.

[PIN86] Piney, C. (1986), 'CABAN (Frigate) Network service description', CERN/DD/COMS-REQ/G4.

[SEG86] Segal, B., O'Brien, R.P. (1986), 'Network information request for CERN TCP/IP network CERNIP', CERN/DD/COMS-REQ/G9.

EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH

Report over the use of IPC on the APOLLO

Author: B. Detrembleur
Date: 28 Nov 86
Version: 1
Status: Final

Use and test of IPC for COMS project

B.Detrembleur[1)]

Geneva, 28 November 1986

Abstract: This report will make a general presentation of IPC, present some typical algorithms , and finally give some tricks for the use.

---

1) I.Informatique

1. General presentation of IPC.

InterProcess Communication facilities (IPC) is one of the most important parts of UNIX BSD4.2. The basic building block for communication is the socket."A socket is a endpoint of communication on which some primitives can be used. Each socket in use has a type and one or more associated processes." [1] Once created within a communication domain, a socket can be named and connected to another one to send or receive data. At the end of the communication, the connection can be closed and the sockets discarded.

1.1 Communication domains.

The properties of processes communicating through sockets within the same domain are common. The communication domain is an abstraction to bundle these properties. For IPC, there are two communication domains : UNIX domain and INTERNET domain. One of the properties of a domain, is the way the sockets are named: in the UNIX domain, the sockets are named with a UNIX path name : "/DEV/FOO", for example. In fact, we won't speak longer about the UNIX domain because the implementation is very bad on the apollo.

1.2 Socket types.

The type of a socket is defined according to the properties visible to the user. Process are presumed to communicate only between sockets of the same type. Three types of sockets are available.

- **Stream socket:** is a bidirectional, reliable, sequenced and unduplicated flow of data without record boundaries. Once connected, a pair of socket provides an interface nearly identical to the UNIX pipe.
- **Datagram socket:** support bidirectional flow of data which is not promised to be sequenced, reliable or unduplicated.
- **Raw socket:** provides users access to the underlying communication protocols which support socket abstractions.

1.3 Socket creation.

To create a socket, the SOCKET system call is used :

s= socket(domain,type,protocol);

With this call, a socket is created in the specified domain and of the specified type. The user can give a zero value for the protocol (usual way) so the system select an appropriate protocol which may be supported with the requested socket type. Let 's have a look for the call to create a Stream socket within the internet domain:

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet./in.h>
#include <netdb.h>

int s;
```

```
if ( (s=socket(AF_INET,SOCK_STREAM,0)) < 0) perror("socket");
```

Where AF_INET is a predefined constant for the internet domain and SOCK_STREAM a constant for the stream socket. Perror is a system call printing a status sentence of the following form:
socket: protocol not supported !

Return value: A -1 is returned if an error occurs, otherwise the return value is small integer referencing the socket.
Errors: See [2] to have a list of errors.

1.4 Binding names.

Once a socket is created, processes must have a way to reference them otherwise no messages can be send to it. The bind call is used to assign a name to a socket.

bind(s,name,namelen);

The bound name is a variable length byte string interpreted by the communicating protocol. Its interpretation may vary from communication domain to communication domain. For the Internet domain, the names contain an internet address and a port number.

```
struct sockaddr_in sin;

int t;

if (bind(t,(char*) &sin,sizeof(sin)) < 0) perror("bind");
```

The structure sockaddr_in contain a field for the family, one for the address of the foreign host and one for the port. (See in chapter 2 how to fill them).

Return value: A zero value is returned if the bind is succesfull. A -1 value indicates an error.

1.5 Connection establishment.

With a bound socket, it's possible to rendezvous with an unrelated process. Usually, this operation is asymmetric with a process "client" and a process "server". The client ask for the connection to the server process. The server is during this time passively listening on its socket.

connect (s,"server name",sizeof("server name"));

S is the socket asking for the connection and server name is the name of a socket of the same type and the same domain which is asked. So for the internet domain, the call is :

```
struct sockaddr_in sin,dst;
int s,t;

      .....
if (connect(s,(char*) &sin,sizeof(sin)) < 0) perror("connect");
```

Return value: If the connection succeeds, then a zero value is returned. Otherwise a -1 is returned.
Errors: see [2] to have a lists of errors. For the server, the listen and the accept calls are used.

listen(s,backlog);

S is the socket on which the server is listening and backlog specify the maximum number of outstanding connections which may be queued awaiting acceptance by the server process. If a connection request occurs when the queue is full, the request will not be refused but ignored, so this give time to the server to make room in his queue while the client ask for connection again. The **MAXIMUM** backlog is 5. For internet domain, the call is :

listen(s,5);

After the listen call, the server is in listening state. He can now accept the connection with the accept call.

```
fromlen = sizeof(from);
snew = accept(s,&from,&fromlen);
```

A new descriptor SNEW is returned on receipt of a connection (along with a new socket). The new socket has the same properties than s but it's not possible to accept new connections on snew. The result parameter FROM contains the address of the calling socket. If the client's name is not of interest, the FROM parameter may be zero. The value result parameter FROMLEN is initialized by the server to indicate how much space is associated with FROM, then modified on return to indicate the true size of the name. There is no way for a process to indicate it will receive connection only from individuals. So it's the user responsability to look the name of the socket calling and to close down connections if it doesn't wish speak to him.

Return values: The call return -1 on error. If it succeeds it returns a non negative integer which is a descriptor for the accepted socket.

1.6 Data transfer.

When the connection is established, data flow may begin. For connected sockets, the calls are:

send(s,buf,sizeof(buf),flags);
recv(s,buf,sizeof(buf),flags);

The flags may be non zero if one the following options is required:

SOF_OOB : send/receive out of band data.
SOF_PREVIEW: look at data without reading.
SOF_DONTROUTE: send data without routing packet.

Out of band data is a logically independent transmission channel associated with each pair of connected stream sockets. Out of band data is delivered to the user independently of normal data along with the SIGURG signal. The SIGURG signal is associated with the existence of an urgent condition. So, for out of band datas, a logical mark is placed in the data stream to indicate the point at which the OOB data was sent. When SOF_PREVIEW is specified with a recv call, any data is returned to the user but treated as still unread. The DONTROUTE option is not interesting for normal users.

For the internet domain, the call is :

```
cc = send(t,buf,sizeof(buf),0);
cc = recv(t,buf,sizeof(buf),0);
```

Return value: cc is the number of sended or received characters. On error, a −1 is returned.

1.7 IPConnection closing.

Once a socket is now longer of interest, it may be discarded by applying a close to the descriptor.

close(s);

If data is associated with a socket which promises reliable delivery (stream socket) when a close take place, the system will continue to attempt to transfer the data. It will be discarded after a long period if the data is still not delivered. If the user have no use of any pending data, it may perform a shutdown on the socket prior to closing it.

shutdown(s,how);

where how is 0 if the user is not interested in reading data, 1 if no more data will be sent or 2 if no data is to be sent or received.

1.8 I/O multiplexing.

It's possible to multiplex i/o requests among multiple sockets or files using the select call.

select (nfds,&readfds,&writefds,&exceptfds,&timeout);

Readfds, writefds and exceptfds are three bit masks. Select examines the i/o descriptors specified by these bit masks to see if they are ready for reading, writing or have an exceptional condition pending. In fact the third bit mask is not implemented. Bit masks are created by or−ing bits of the form "1 < <fd". That is, a descriptor fd is selected if a 1 is present in the fd'th bit of the mask. The parameter NFDS specifies the range of the file descriptors (one plus the value of the largest descriptor) specified in the masks. If timeout is a non zero pointer, it specifies a maximum interval to wait for the selection to complete. If timeout is a zero pointer, the select blocks indefinitely. To affect a pool, the timeout argument should be non−zero, pointing to a zero valued timeval structure. Any of the masks may be given as zero if no descriptors are of interest.

Return value: Select returns the number of descriptor which are contained in the bit masks, or −1 if an error occured. Zero is returned if the time limit expires.

2. Typical algorithms for IPC.

2.1 Client program.

Let's first have a look to typical algorithms. For these algorithms, internet sockets are used on different machines: Apollo Coms (apo – coms) and Vax Unix (Priam).
The first algorithm is for a client process :

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>



extern errno;

struct sockaddr_in sin = { AF_INET };
struct sockaddr_in dst = { AF_INET };

struct hostent  *hp;
struct servent  *sp;
int cc,ns,backlog;
int s,t;
int *fromlen;
int flags,len;



main()

{

char buf[1000];

int fid;
int i,j,temp,nlu;

  /* Preparation of the message */


   nlu = read(fid,buf,1000);

   sp = getservbyname("coms","tcp");

/* By this call a structure containing the port dedicated for the
   coms project is given in the sp structure.
!! attention !! The port number on priam for coms is presumed to be
   same than on the apollo .                                    */


   hp = gethostbyname("priam");
```

```
/* By this call a structure containing the internet address of Priam
   is filled.                                                    */

   dst.sin_family = AF_INET;
   dst.sin_addr.s_addr = hp->h_addr;
   dst.sin_port = ntohs(sp->s_port);
```

> !! **LOOK OUT** !! NTOHS is a byte swapping routine (network to host short). These routines are provided because the operating system expects addresses to be supplied in network order. On a VAX, or machine with similar architecture, this is usually reversed. Consequently, programs are sometimes required to byte swap quantities.

```
   hp = gethostbyname("apo-coms");

   sin.sin_family = AF_INET;
   sin.sin_addr.s_addr = hp->h_addr;
   sin.sin_port = sp->s_port;


/* Creation of the socket */

if ( (t = socket(AF_INET,SOCK_STREAM,0)) < 0) bomb("socket");


/* Binding of a name to the socket */

if (bind(t,(char*) &sin,sizeof(sin)) < 0) bomb("bind");


/* Connection of the socket */

if (connect(t,(char*) &dst,sizeof(dst)) < 0) bomb("connect");


/* The message can now be send */

if ((cc = send(t,buf,nlu,0)) < 0) bomb("send");


/* The connection is closed and the socket discarded */

if(shutdown(t,2) < 0) bomb("shutdown");
close(t);
exit();
}
```

```
bomb(s)
char *s;
{
perror(s);
}
```

2.2 Server program.

Let's now have a look to the server program.

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>



extern errno;

struct sockaddr_in dst = { AF_INET };
struct sockaddr_in adr = { AF_INET };

struct hostent  *hp;
struct servent  *sp;
int cc,ns,backlog;
int s,t;
int *fromlen,addrlen;
int flags,len;



main()

{

char buf[1000];

int fid,i,cc;


   sp = getservbyname("coms","tcp");

/*The way to prepare the addresses are the same than in the previous
   program. */

   hp = gethostbyname("apo-coms");

   dst.sin_family = AF_INET;
   dst.sin_addr.s_addr = hp->h_addr;
   dst.sin_port = sp->s_port;



if ( (s = socket(AF_INET,SOCK_STREAM,0)) < 0) bomb("socket");

if (bind(s,(char*) &dst,sizeof(dst)) < 0) bomb("bind");

/*The socket is put in a listening state,the connection requests can
     be sent. */
```

```
listen(s,5);

 /* The loop is infinite */

for (;;)
{
     addrlen = sizeof(addr);
     ns = accept(s,&addr,&addrlen);

/* The accept blocks until the arriving of a connection request,
  when the request arrive, the message is waiting in the ns socket*/

     if (ns < 0) bomb("accept");

/* A fork process is created to take care of the message */

     else  if (fork() == 0) fils1(ns,addrlen);
     close(ns);

}


if(shutdown(s,2) < 0) bomb("shutdown");

close(s);
exit();


}

bomb(s)
char *s;
{
perror(s);
};

fils1(n,len)
int n;
int len;
{
   int cw,i;

recw = recv(n,buffer,1000,0);

/* ..... Make something with the message ..... */

   if(shutdown(n,2) < 0) bomb("shutdown");
   close(n);
   exit(0);
}
```

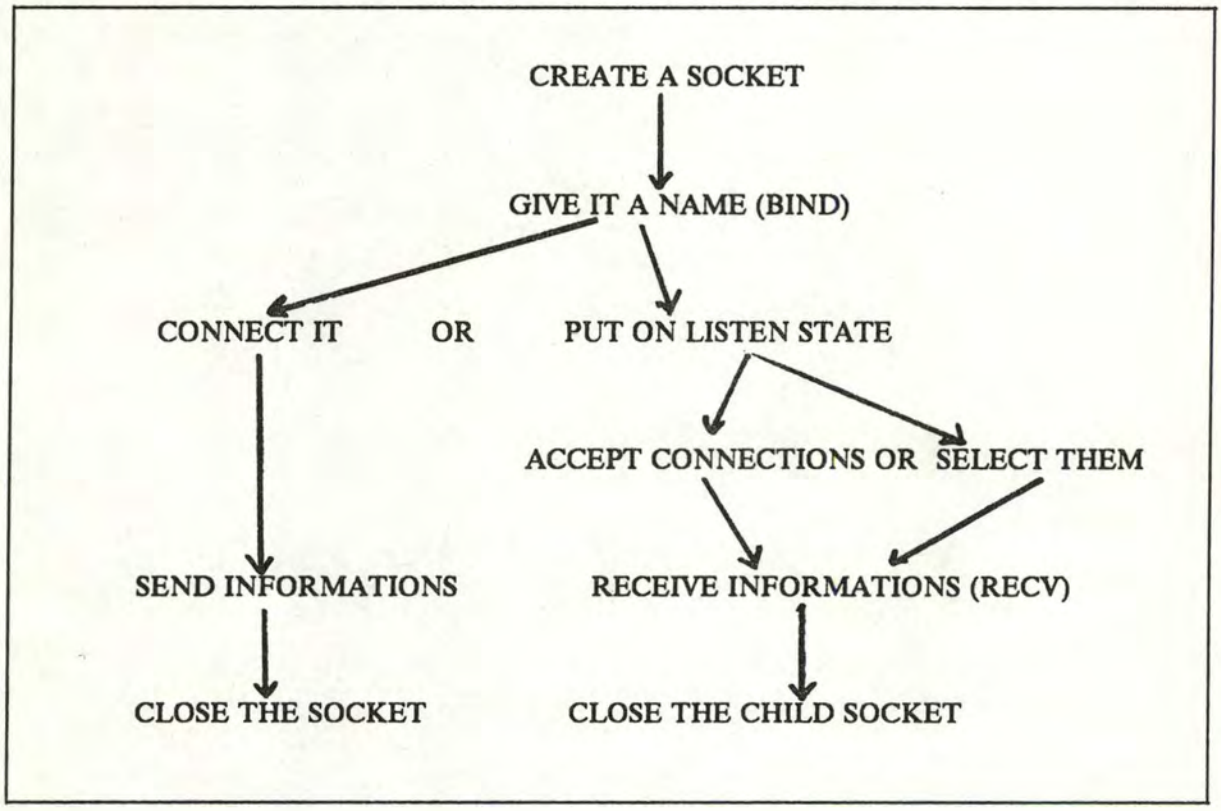
3. Conclusions.

IPC on the Apollo seems to follow the standard except for the UNIX sockets which are very difficult to use. Remember the main points which are not very clear in the documentation:

1. To speak between differents machines, you must know the port number of the socket you want to join. The best to know it seem it to assign the same port number on the two machines for an distributed application. For example: The messages for the display manager on the Apollo from the service manager on the vax will follow this way:
   The VAX send on port NTOHS(NUM).
   The APOLLO listen on port NUM.

2. This introduce the second point: never forget the addresses are BYTE SWAPPED on VAX.

Having these two points in mind, you just have to use the primitives in a logical chronology:



Contents

References

[1] Leffler, S.J. Fabry, R.S. and Joy, W.N. (1983),'A 4.2bsd Interprocess Communication Primer',DRAFT at July 27,1983.

[2] Unix authors, (1983),'Unix programmer's manual', available in bsd 4.2 release documentation of 7 july 1983.

EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH

Report over the use of
DIALOGUE on the APOLLO

Author: B. Detrembleur
Date: 14 Nov 86
Version: 1
Status: Draft

Use and test of DIALOGUE for COMS project

B.Detrembleur[1)]

Geneva, 14 November 1986

Abstract: This report will make a general presentation of Dialogue, present some typical algorithms , and finally give some tricks for the use.

---

1) I.Informatique

1. General presentation of Dialogue.

Dialogue is a tool for defining the interface to an application program and specifying how the interface should be presented to the users of the application . The Dialogue part of a program has an user interface and an application interface (see figure 1).

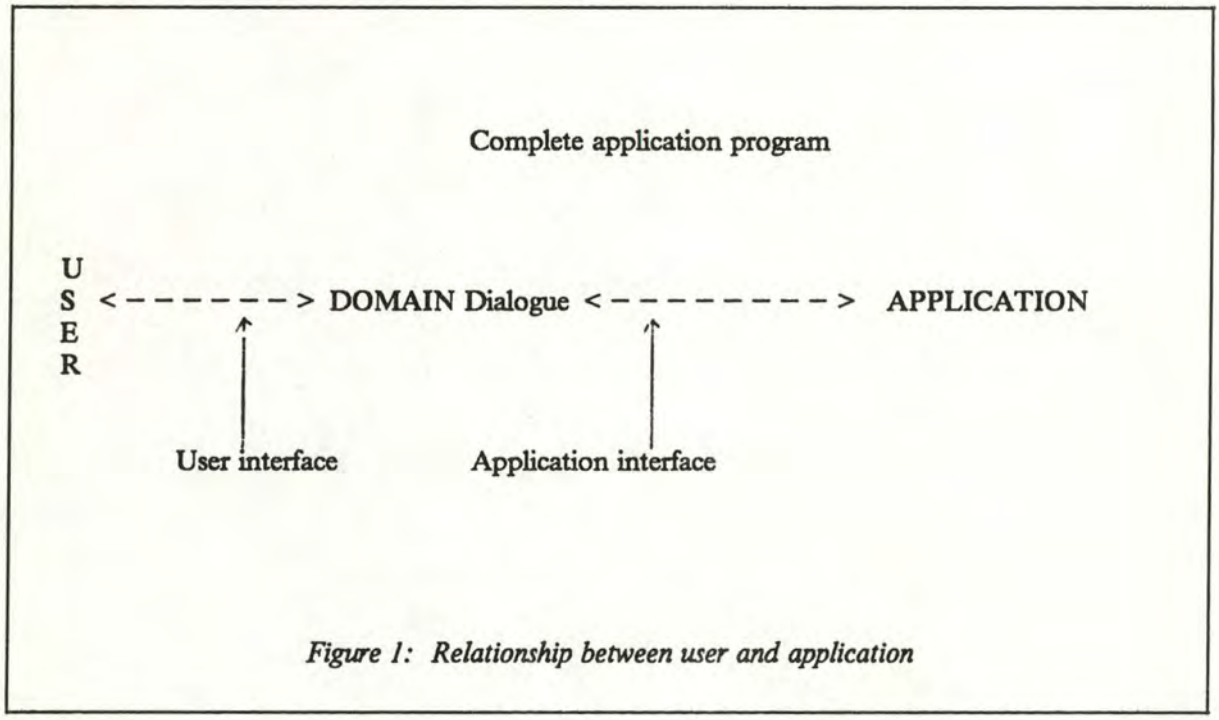


Figure 1: Relationship between user and application

1.1 Files of a dialogue application.

For an application using dialogue, there are three differents files.

- A file with the dialogue commands called FILENAME.DPS .
- A file with all the routines of the application.
- A file with the main program .

1.2 Connections between the files of a dialogue application.

The description file (FILENAME.DPS) is compiled with the dialogue translator. This translator produce a binary description file (FILENAME.DPD) and an application specific insert file. The application specific insert file is included in the application with Standards Dialogue insert file. The application is then compiled and binded. (See figure 2) .

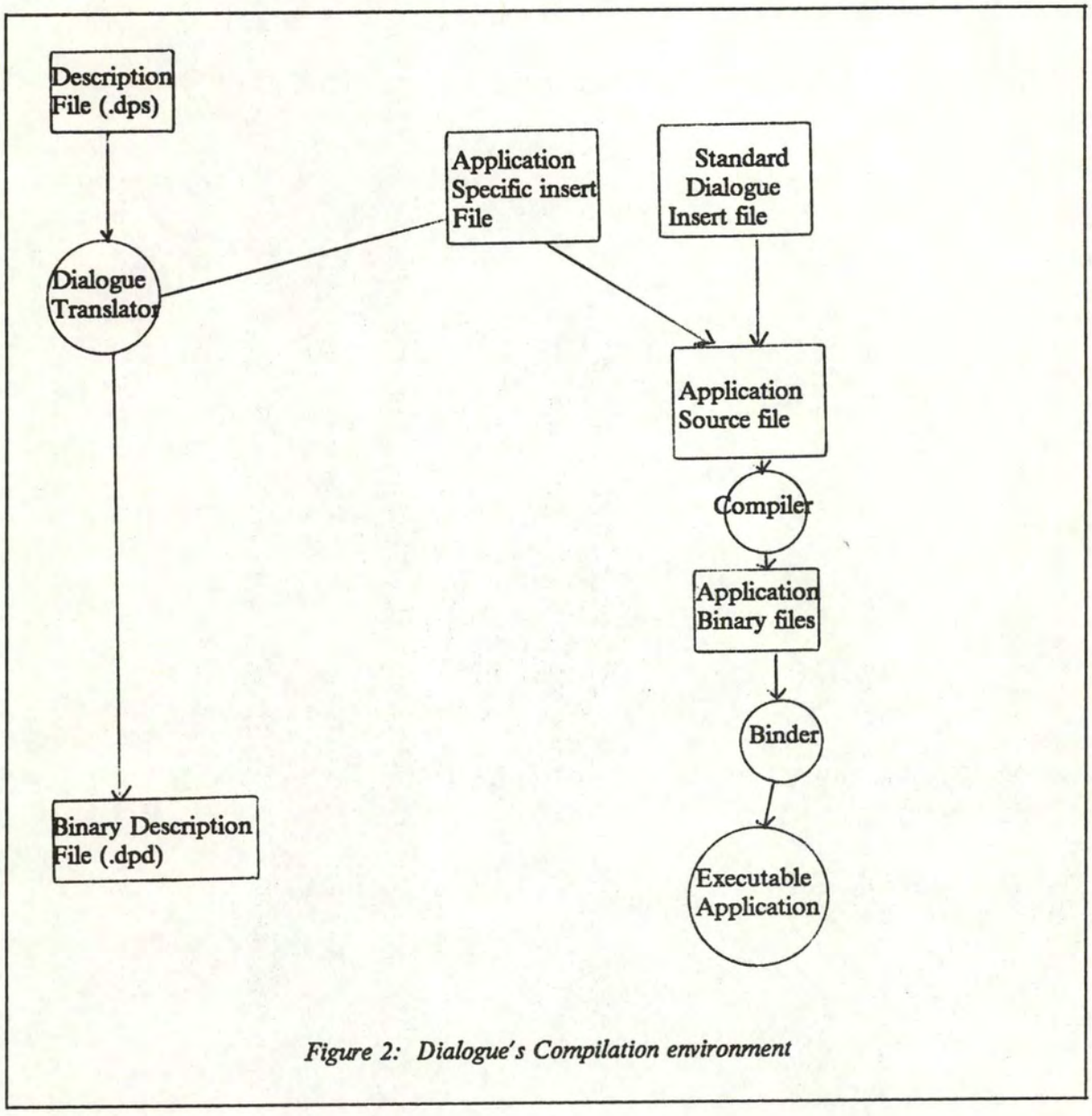


Figure 2: Dialogue's Compilation environment

1.3 The description file.

The description file is divided in two part :

- **The Application – Interface.**
- **The User – Interface.**

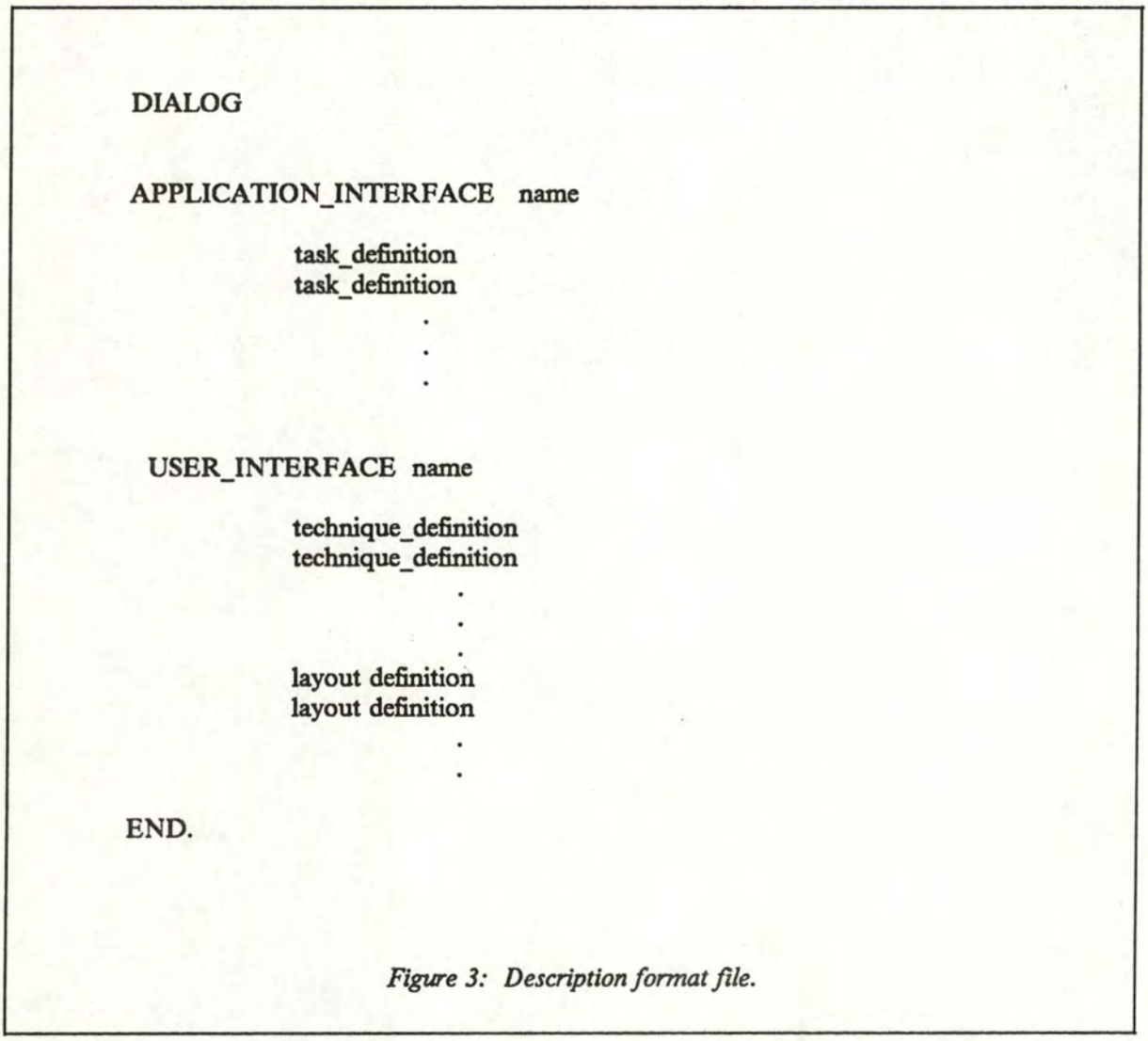


Figure 3: Description format file.

1.3.1 The application – interface.

The application – interface is composed of tasks which denote things a user can do to affect the application such as provide data or make a request. So a task represents an activity.To satisfy the application's needs for input regarding its activities, each task defines a valid data type that the user can provide. For example to add two numbers, the application requires numerical data.

1.3.1.1 Task.

The tasks and data types are :

- **Bool** Boolean data.
- **Enum** (*) enumerated data.
- **GMR** (*) GMR input.

- **GPR** GPR input.
- **Int** integer data.
- **Msg** multiline text string.
- ***Null (*):*** null data (no data).
- **Real** real numbers
- ***Set:*** set data
- **String** single line text string.

The types with an asterisk are used often.

Enumerated data consists of a list of choice that the application designer defines in the task, from which the user will select one. GMR and GPR data consist of button and/or mouse inputs,keystrokes and locator movements. Null task is used in cases the application can act on a user's request without retrieving any data (such as a request to leave the application) . The task can either call the application, activate or deactivate a group of tasks, or return to the application.

1.3.1.2 The task's syntax.

Figure 3 illustrates the format of a description file. Task definitions are placed under the heading APPLICATION_INTERFACE. Figure 4 illustrates task – definition syntax.

```
name := type:
        event = > < action_list > ;
            .
            .
            .
        attribute = value ;
            .
            .
            .
end
```

Figure 4: A task definition.

Name/Type declaration: Each task has an identifier and type. This identifier is bound to an integer value in the application – specific insert file. Identifiers are also used in techniques definitions to refer to a task.

Event/Action statements: A task event occurs when the user provides valid input to the task's technique. The task receive the input and takes whatever action is specified in its action list. Task events are :

- **Comp** which indicates that the user input was received and the task therefore completed.
- **Help** which indicates that the user requires help.
- **GPR and GMR input events** which are : buttons, keystroke, locator, locator_stop, entered_window, left_window.

In response to an input event, a task can take zero or more actions. The actions are :

- **Call** an application subroutine.
- **Deactivate** a task or a task group.
- **Activate** a task or a task group.
- **Return** control to the application.

Attribute/Value statements: provide further informations that the task needs in order to function.

- **Identifiers:** for listing things, such as choices in an enumerated task.
- **Integers and real numbers:** for delimiting minimum and maximum data range.
- **Keyset:** for defining keys to use in a GMR or GPR task.
- **Size:** for specifying the size of a bitmap for GMR or GPR task.
- **Strings:** for supplying text.

End statement: This ends the definition.

```
APPLICATION_INTERFACE dmtest

        quit:= NULL:
              COMP = > <RETURN>
              END

        make_rep := ENUM:
              COMP = > <CALL make_report ACTIVATE quit>;
              CHOICES = (Cernet Ethernet Domain All);
              END
```

Figure 5: Example of a task definition.

1.3.2 The user interface.

The user interface is the part of a dialogue program in which the programmer can define presentation and structuring techniques.

- Presentation techniques are visual devices such as icon or menus. These allow the user to provide input that DIALOGUE can store and in which it can notify the application.
- Structuring techniques combine presentation techniques into arrangements, such as row or column, to form a screen layout.

Presentation technique :

```
quit_icon := ICON:
TASK      = quit;
SHAPE     = ROUNDED;
STRING    = "Exit";
HELP_TEXT = "Select Exit to leave the program";
END
```

(The words in uppercase are the keywords of dialogue)

Structuring techniques :

```
Line_with_menus := ROW:
HELP_FILE       = "//zeta/user/bernard/help";
ORIENTATION     = VERTICAL;
CONTENTS        = ( quit_icon
                    erase_icon
                    draw_menu );
END
```

Figure 6: Technique presentation and structuring techniques

1.3.2.1 Techniques.

Techniques specify how the user can enter input to a task. Presentation techniques define visual mean of displaying task. The user−interface portion of a description file contains presentation techniques for tasks defined in the application portion. There can be multiple techniques per task,except for GMR or GPR tasks, which can only have one technique apiece. So each technique MUST be connected to one task and each task MUST have at least one technique. The different techniques are :

- **Bool_field** for entering Boolean data.
- **Enum_field** for selecting one or several items.
- **Int_field** for entering integers.
- **Real_field** for entering real numbers.
- **Set_field** for selecting several items.
- **String_field (*)** for entering a single line of text.
- **Display_text (*)** for displaying multiple lines of read−only text.
- **Graphics_area (*)** for entering GMR or GPR input.
- **Icon (*)** for generating a event with no data.
- **Menu (*)** for selecting one or several items.
- **Scrollbar** for scrolling a menu or switch technique.
- **Switch** for cycling through choices.

The asterisk means that this technique is used often.

1.3.2.2 Structuring techniques.

Structuring techniques allow the user−interface designer to design a screen layout by grouping and arranging presentation techniques (or other structuring techniques).

Their types are :

- **Row** organizes techniques into row , it can be oriented vertically to organize techniques into column.
- **Oneof** contains several techniques, but display only one at a time.
- **Window** defines an initial window in which the application runs and a technique to run the window.
- **Popup** defines a rectangular area that can be superposed over the initial window.
- **Space** provides space between techniques.

1.3.2.3 Syntax of techniques and structuring techniques.

Figure 6 illustrates technique definition syntax. Technique definition consist of the following: **Name/ Type statement:** Each technique has an identifier and type. Structuring techniques use identifiers to designate techniques they control. The types were listed above. **Event/Action statement:** A technique event occurs when the user provides input to a technique. The technique receives the event and takes whatever action that is listed in his action list. Techniques event are :

- **A list of keys or mouse buttons** that the user can press while the cursor is into a technique.
- **Enter** which means that the user moved the cursor into DOMAIN/Dialogue window.
- **Leave** which means that the user moved the cursor out of a popup or out of a DOMAIN/Dialogue window.
- **Select** which means that the user selected an icon or selected something from a menu.

```
name := type:
        event => <action_list> ;
           .
           .
           .
        attribute = value ;
           .
           .
           .
end
```

Figure 7: A technique definition.

In response to an event, a technique can take zero or more actions. Actions are listed below but for more indications about the actions, please refer to the Dialogue Manual page 6−37.

- **Accept**
- **Backspace**
- **Be_current_child**
- **Char_del**
- **Cursor_left or Cursor_rigth**
- **Help**
- **Insert_toggle**
- **Line_del**
- **Next_field**
- **Popdown**
- **Scroll_down, left, left_justify, left_unit, rigth, rigth_justify, rigth_unit, up**
- **Select**
- **Show**

Attribute/Value statement: let the user modify a technique's presentation and functions.

- **Entries:** for naming things such as the entries that will be displayed in a menu.
- **Identifiers:** for specifying things such as the shape or orientation of a technique.
- **Integers and real numbers:** for specifying size.
- **Keyset:** for defining keys to use in a technique.
- **Size:** for specifying technique sizes.
- **Strings:** for supplying text.

```
USER_INTERFACE dmtest

        make_rep_menu := MENU:
                TASK = make_rep;
                BACKGROUND = OFF;
                SHAPE = ROUNDED;
                TITLE_STRING = "Make a Report";
                ENTRIES =
                        ("Cernet"      => cernet
                         "Ethernet"    => ethernet
                         "Domain"      => domain
                         "All"         => all );
                    END
```

Figure 8: Example of a menu definition.

1.4 File with all the routines of the application.

A file will regroup all the routines of the application. It means all the routines called in a COMP statement of a task and all the internals routines. All these routines can use the Dialogue System Calls. The user can find these system calls in section 7 of the Dialogue manual. All the system calls have the same form: DP_$XXXXXXX. The different calls are :

- Calls to initialize and terminate Dialogue. NOTE that these calls MUST be used with GPR and GMR, so you don't have to initialize the graphics.
- Calls to wait for an event. Only these system calls can be used for a wait in dialogue.
- Calls to retrieve or to pass application specific data.
- Calls to activate or deactivate task groups.
- Calls to notify user of condition or error.
- Calls to retrieve or to pass information to a task.
- Calls to handle ENUM.
- Calls to handle SET.
- Calls to handle GMR.
- Calls to handle GPR.

1.5 File with the main program.

This file is very similar for all the Dialogue application so we can give a weft for it (see figure 9). The aim of this main program is to initialize dialogue, activate some tasks, wait for an event on input and terminate dialogue.

1.5.1 Initialize Dialogue.

To initialize Dialogue, the user can use two ways:

- **DP_$INIT(...,"filename.dpd",...)** is used during prototyping, because doing so , the user doesn't have to recompile and rebind all the application every time he wants to change the USER_INTERFACE. He just have to retranslate the file FILENAME.DPS to create a new binary description file.
- **DP_$INIT_FROM_MEMORY(...,&dp_filename_heap,...)** is used when the application is ready, the user must bind the object module ("FILENAME_DPD.BIN") with the application. So the user doesn't need worry about keeping files together or ensuring that stray versions of the description are compatible with the application. The compatibility is checked with the DP_$KEY parameter of DP_$INIT.

1.5.2 Initialize some tasks.

To initialize some tasks , the system call is DP_$ACTIVATE(...). It can be useful to group the tasks in task group, so they can be activated together. To deactivate tasks , use DP_$DEACTIVATE(..).

1.5.3 Wait for an event on input.

The call DP_$EVENT_WAIT(...) will wait for an event on input. It is an infinite loop until the user asks for a task in which the COMP clause is COMP = > <RETURN>.

1.5.4 Terminate the application.

To terminate the application, DP_$TERMINATE(...) is used.

```
/* insert the include files */

main()

{
        /* initialize application data  */
                                .
                                .
        /* initialize Dialogue */

        DP_$INIT(unit_or_pad,dsc_file_name,dsc_file_name_len,
                                entry_vector,dp_key,status);

        /* set default or initial value */

        DP_$type_SET_VALUE(task_id,value,status);
                                .
                                .

        /* activate a task group */

        DP_$TASK_ACTIVATE(task_id,status);

        /* wait for an event on input */

        DP_$EVENT_WAIT(task_id,event_id,status);

        /* exit Dialogue */

        DP_$TERMINATE(status);

}
```

Figure 9: Typical main program.

2. Example of the use of Dialogue.

The specification of this example is to present in a dialogue menu the differents segments of a GMR metafile which contains the representation of elements of networks. The user can choose to see a node, a host, a host switching node, a modem, a patch panel, a multiplexer, a ring, a bus or a subnet.

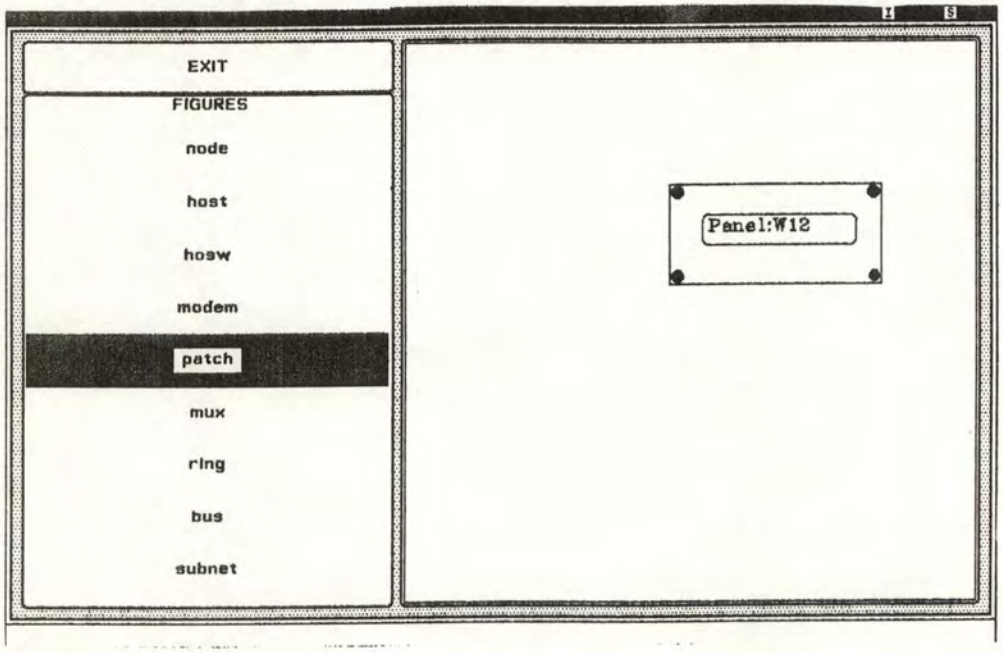


Figure 10: Screen of the example.

2.1 The description file

The description file is the first file you must write for the application. Let's have a look to this file which produce the screen in figure 10.

```
DIALOG

APPLICATION_INTERFACE mod4

    workarea_task := gmr:          { definition of the graphic area}
            LOCATOR   = >  <CALL work_area_locator>
            END
```

```
quit := NULL:   {definition of the exit task }
        COMP = > <RETURN>
        END


draw_object := ENUM:     { set_draw_object is called when this task}
        COMP = > <CALL set_draw_object> ; { is completed }
        CHOICES = (node host hosw modem patch mux ring bus subnet)
        END




USER_INTERFACE mod4

%INCLUDE "//epsil/sys/ins/dialog_user.ins.dps"


quit_icon := ICON:
      TASK = quit ;
      STRING = "EXIT" ; { the string EXIT appears in the icon }
      HELP_TEXT = "Select EXIT to leave the DRAW program." ;
      SHAPE = rounded
      END


draw_object_menu := MENU:
      TASK = draw_object ;
      BACKGROUND = off ;
      SHAPE = rounded ;
      TITLE_STRING = "FIGURES" ; { title of the menu }
      ENTRIES =
            ("node"     = > node    { definition of the string }
             "host"     = > host    { for the different choice }
             "hosw"     = > hosw
             "modem"    = > modem
             "patch"    = > patch
             "mux"      = > mux
             "ring"     = > ring
             "bus"      = > bus
             "subnet"   = > subnet ) ;
      HELP_TEXT = "Select a figure and draw it as follows:"
            &"NODE:    design of a node."
            &"HOST:    design of a host."
            &"HOSW:    design of a host switching node."
            &"MODEM:   design of a modem."
            &"PATCH:   design of a patch panel."
            &"MUX:     design of a multiplexor."
```

```
                &"RING:     design of a ring infrastructure."
                &"BUS:      design of a bus infrastructure."
                &"SUBNET:   design of a subnet.";
        END


workarea_tech := GRAPHICS_AREA:
        OUTLINE = on ;
        TASK = workarea_task ;
        SHAPE =  rounded ;
        SIZE = ((600 600) (600 600) (600 600))  pixels
        END
```

This workarea_tech will give the screen in figure 11.

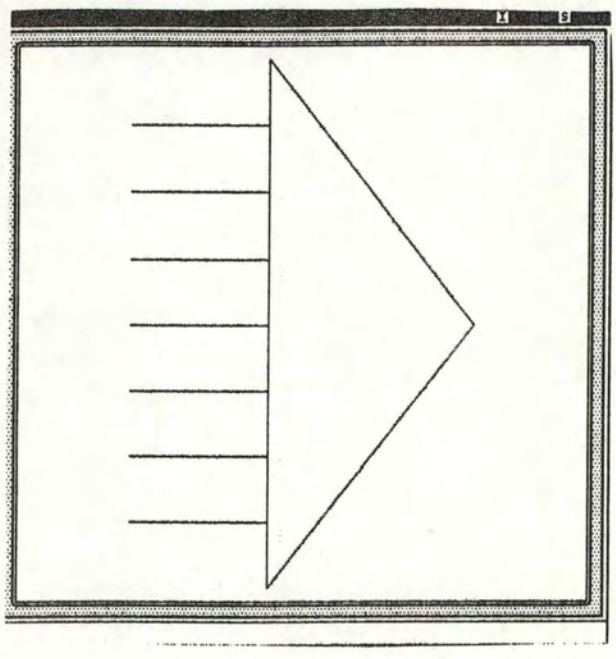


Figure 11: Workarea_task technique.

```
column_with_menus := ROW:
        ORIENTATION = vertical ;
        BACKGROUND = gray ;
```

```
        CONTENTS = (quit_icon
                    draw_object_menu)
        END
```

The column_with_menus technique give a row with the quit_icon and the draw_object_menu.

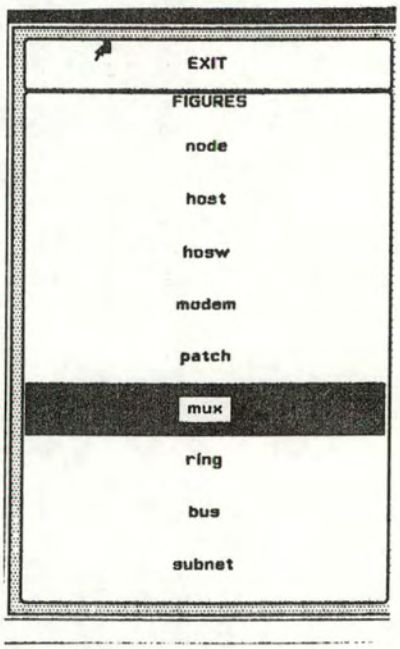


Figure 12: Column_with_menu technique representation.

```
row_with_menus_and_graphics_area := ROW:
        BACKGROUND = gray ;
        BORDER_WIDTH = 10 ;
        DIVISION_WIDTH = 7 ;
        OUTLINE = on ;
        SHAPE = square ;
        CONTENTS = (column_with_menus
                    workarea_tech)
        END
```

The column_with_menus and the workarea_tech are put together in the row_with_menus_and_graphic_area, this give the screen in figure 12.

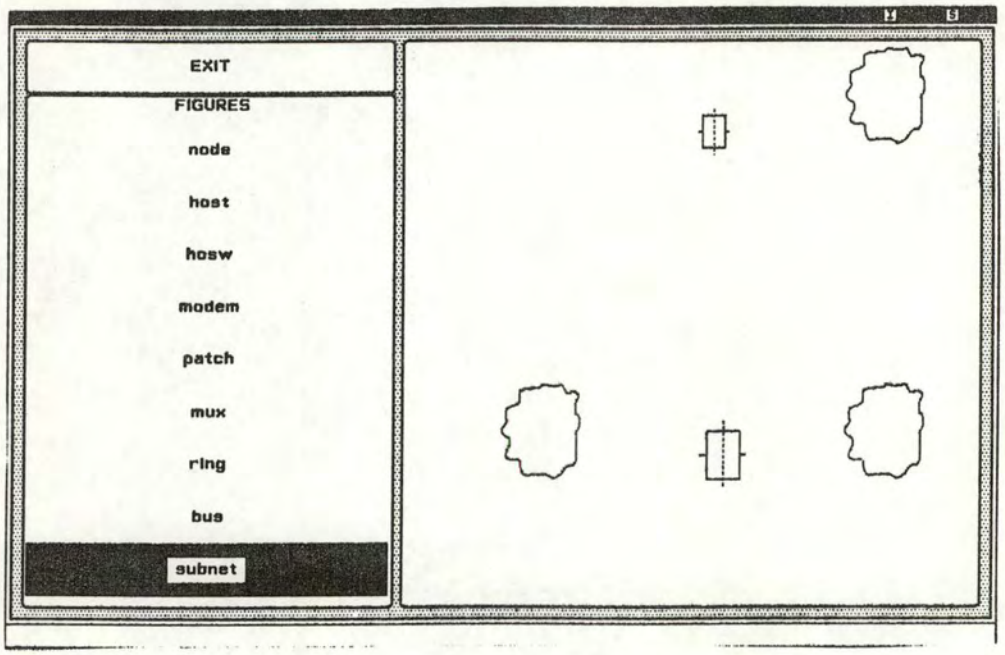


Figure 13: Complete screen.

```
std_window:
        HELP_FILE = "/domain_examples/dialog/draw.hlp" ;
        CONTENTS = row_with_menus_and_graphics_area
        END

END.
```

2.2 The main program.

In fact, the structure of the main program is nearly the same for all applications. So refer to the comments in the program for some particularities.

```
#nolist
#include <stdio.h>
#include "//epsil/sys/ins/base.ins.c"
#include "//epsil/sys/ins/error.ins.c"
#include "//epsil/sys/ins/gpr.ins.c"
#include "//zeta/sys/ins/gmr.ins.c"

#list


/*_____________________________________________________________________
  |                                                                   |
  | Standard DIALOG Package insert file.                              |
  |___________________________________________________________________|
*/
#include "//epsil/sys/ins/dialog.ins.c"

/*_____________________________________________________________________
  |                                                                   |
  | Insert file generated by the DIALOG translator for this application.  |
  |___________________________________________________________________|
*/
#include "mod4.ins.c"


extern short                current_draw_object ;
gm_$point32_t    bitmap_size = {500,500};

main ()
{
        status_$t               status,st ;
        dp_$task_id              task ;
        dp_$event_id             event ;

        dp_$boolean              unobs ;
        int                 file_id;
/*_____________________________________________________________________
  |                                                                   |
  | Initialize DIALOG. Pass the name of the description file to use       |
  | (mod4.dpd).                                                       |
  |___________________________________________________________________|
*/


    dp_$init(stream_$stdout,"mod4.dpd",(sizeof ("mod4.dpd") - 1),dp_$mod4_entry_vector,
                                                        dp_$mod4_key, status) ;
```

```
/* Open the graphic metafile that contains the pictures oh the elements */

    gm_$file_open("gmben2",(short)6,gm_$r,gm_$1w,file_id,st);



/*_____________________________________________________________________
  |                                                                    |
  | Initialize menu choices.                                           |
  |____________________________________________________________________|
*/
    dp_$enum_set_value (draw_object, current_draw_object, status);

    dp_$task_activate (dp_$all_task_group, status) ;


    if (status.all != 0)
        error_$print (status) ;

/*_____________________________________________________________________
  |                                                                    |
  | Leave control with DIALOG until the user exits.                    |
  | Acquire and release the display if the program uses GPR or GMR.    |
  |____________________________________________________________________|
*/


    gm_$viewport_set_refresh_state(gm_$refresh_update,st);

    unobs = gpr_$acquire_display (status) ;


    dp_$event_wait (task, event, status) ;

    gpr_$release_display (status) ;

    dp_$terminate (status) ;

    return ;
}
```

2.3 The routines file.

The routines file is (off course) application dependent. So it's not useful to present here an example. But it's interesting to see how dialogue give arguments to the application.

```
/* This function is declared VOID, because it has been declared
     by dialogue in the insert file.                          */

void set_draw_object  ( task_id, event_id )

/* The argument TASK_ID contains the number of the calling task ,
and the EVENT_ID the number of the calling event. */

dp_$event_id   *event_id ;      /* input  */
dp_$task_id    *task_id ;       /* input  */

{   short file_id;

      dp_$enum_get_value (*task_id, current_draw_object, st);

/* With the task_id , we can ask dialogue to know which
current_draw_object is asked by the user. */



      switch (current_draw_object)
      {

            case node:
                  des_node();
                  break ;

            case host:
                  des_host();
                  break ;

            case hosw:
                  des_hosw();
                  break ;

            case modem:
                  des_modem();
                  break ;

            case patch:
                  des_patch();
                  break ;

            case mux:
                  des_mux();
                  break ;
```

```
            case ring:
                    des_ring();
                    break ;

            case bus:
                    des_bus();
                    break ;

            case subnet:
                    des_subnet();
                    break ;


    }


    return;

}
```

3. Some tricks to use DIALOGUE.

Using DIALOGUE, you will discover some difficulties, a part of these one are listed above.

3.1 INLIB.

Each time you want to use DIALOGUE in a shell, you have to make an INLIB. This command give the shell an access to the system library. For example, if you use DIALOGUE with GMR, you have to type :

```
INLIB /LIB/GMRLIB
INLIB /LIB/DIALOGLIB
```

It 's possible to introduce these commands in profile used each time you create a new shell.

3.2 Bind a DIALOGUE program.

To bind a DIALOGUE application, you can use the following format :

```
BIND xxxxxxx.bin @
     yyyyyyy.bin @
     zzzzz_dpd.bin @
     -b ttttt
```

There seem to have some problems if you try to bind the application with the libraries, so with the INLIB, you can use a BIND of this format.

3.3 Writing a DIALOGUE application.

The method to follow to write a DIALOGUE application is :

- Write the application_interface.
- Write a very simple user_interface.
- Write the main program (see chapter 2).
- Write the routines.
- Write the complete user_interface and prototype it.

Contents

Figures